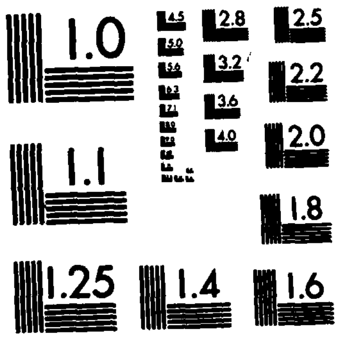
AD-A124 665 A COMPUTER MODEL TO AID THE PLANNING OF RUNWAY ATTACKS 1/2
(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH
SCHOOL OF ENGINEERING H M HACHIDA DEC 82
UNCLASSIFIED AFIT/GOR/OS/82D-6 F/G 1/5 NL


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

A COMPUTER MODEL TO AID THE PLANNING OF RUNWAY ATTACKS

THESIS

AFIT/GOR/OS/82D-6

Howard M. Hachida
Captain USAF

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY (ATC)
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

A COMPUTER MODEL TO AID THE PLANNING OF RUNWAY ATTACKS

THESIS

AFIT/GOR/OS/82D-6 Howard M. Hachida
Captain USAF

AFIT/GOR/OS/82D-6

A COMPUTER MODEL TO AID THE PLANNING OF RUNWAY ATTACKS

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

Howard M. Hachida, B.A.

Captain USAF

Graduate Operations Research

December 1982

Preface

This research topic was suggested by Dr. Edward J. Dunne, Adjunct Professor in the Operational Sciences Department, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH.

I have found the field of runway closure to be very relevant today, both in the probability theory and in the renewed interest in attack strategies. Hopefully, the development of a computer program to aid in planning runway attacks will provide further insights and experience for planners.

I am grateful to my thesis advisor, Dr. Dunne, for the assistance, encouragement and support he provided during this effort. I am also indebted to Lt. Col. James N. Bexfield for his understanding and technical advice.

List of Figures

List of Tables

List of Frequently Used Symbols

AIMLOC(I)	the location of aim point i
CEP	the circular error probable describing the weapon accuracy
DAMRAD	the weapon damage radius
ERUNLE	the effective runway length
ERUNWI	the effective runway width
ETOLEN	the effective MLW length
ETOWID	the effective MLW width
MLW	minimum launch window, the minimum space required for aircraft operations on the runway
NCUTS	the number of cuts required to close the runway
NITERA	the number of iterations for the monte-carlo simulation
NSPW	the number of sections per runway width
NUMAIM	the number of aim points for particular cut
NUMBOM(I)	the number of weapons for aim point i
PC	the probability of a runway cut
P_{CL}	the probability of runway closure
RC	the event that the runway is cut
RUNLEN	the runway length
RUNWID	the runway width
s_i	the individual sections equal to the effective MLW width
TOLEN	the MLW length
TOWID	the MLW width

Abstract

A computer program to aid the planning of runway attacks is developed. Conventional, individually targeted weapons are used against non-reinforced concrete runways. The program has two main sections. The first section evaluates any attack strategy, based on independent cuts along the runway, with each cut specified in terms of number of aim points, number of weapons per aim point, and aim point locations. The second section searches for the "best" strategy which uses the least number of weapons to achieve an overall probability of runway closure equal to or greater than a user specified level.

The program operates in three modes. The mode 1 program returns the fewest number of weapons and the "best" strategy in order to meet or exceed a user defined level of runway closure. Mode 2 allows the user to specify a fixed number of weapons instead of a level of runway closure, and the program returns the highest probability of runway closure and the "best" strategy to use with the fixed number of weapons. Finally, mode 3 allows the user to completely specify a strategy in terms of number of cuts, cut locations, number of aim points per cut, number of weapons per aim point and locations of the aim point, and the program returns the expected probability of runway closure for the user defined strategy.

<u>Contents</u>

Contents

I. Introduction

Background

The airfield complex, normally attacked in hopes of reducing the enemy's air capability, is a popular target for military strategists. Of primary importance are the aircraft because of thier delicate structure, fuel content, and high-explosive weapons. Two factors, however, make aircraft less opportune targets than airfields. First, aircraft parked in shelters are difficult to destroy since the attacker must breach or substantially damage the shelter in order to reach the aircraft. Special weapons, whose accuracy in some cases demand special aerial maneuvers, make attacking aircraft in shelters vulnerable to ground defense systems. Secondly, increases in aircraft performance make aircraft less opportune targets. Today, attacking aircraft can strike, return to base, rearm, and be over the target again in an extremely short time, often shorter than that required to repair the damaged runway before defending interceptors may be launched. Thus, opposition fighters can be effectively eliminated from the first stages of battle without being physically damaged by denying them a launch and recovery surface. The concept that runways . . . are poor targets is no longer valid in the light of the pace of modern warfare and the capabilities of aircraft. (Ref 11)

Currently, development of "better" weapons for runway destruction is the emphasis in weapons development. An example is the French-made Durandal runway attack weapon, which is a parachute-retarded, rocket-boosted, concrete-penetration bomb designed to make craters of runway surfaces. It is now a part of the Foreign Weapons Evaluation Program, and if further testing is successful, deliveries will be scheduled for mid-1983. (Ref 2)

Problem Statement

The following scenario forms the basis for this research. How should a wing commander allocate his resources to effectively close a runway? The runway is a rectangular non-reinforced concrete area for the purpose of launching and recovering fixed-wing aircraft. The runway is closed when sufficient space to launch his aircraft does not exist. This space is generally thought of as a rectangle and will be referred to as the minimum launch window (MLW).

The commander has available only one type of individually targeted conventional runway munition for the mission. The attack consists of cutting the runway at specified locations along the length of the runway. Each cut is a set of aim points located along a line perpendicular to the runway length (see Figure 1). A strategy is a set of completely specified cut locations, aim points per cut and number of weapons per aim point.

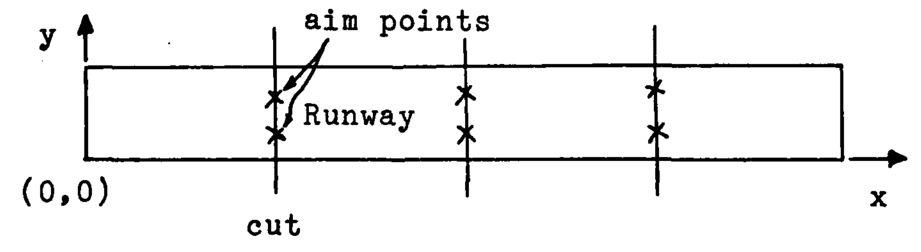


Figure 1. Attacking Strategy

For reference, the origin of the x-y plane will be at the southwest corner of the runway (see Figure 1) and the length of the runway will be on the x-axis. The cut locations are then the distances along the x-axis. The aim point locations are denoted by the cut location and the distance from the edge of the runway.

A review of available literature, discussed in Chapter II, shows that the targeting strategies for runway destruction has not kept up with the development in weapons. There are only a few models to evaluate targeting strategies and none which will find a "best" strategy. Thus, the commander, or any military planner, has to rely on experience or insight to come up with a targeting strategy before using one of the existing models.

This research effort develops a computer program, described in Chapter V, which will find a "best" targeting strategy for a given problem without requiring the commander to come up with a targeting strategy himself. This computer program will aid the commander by answering these three questions:

1. How many weapons does it take to obtain a certain level of runway closure?

2. What is the highest level of runway closure that can be obtained with a fixed amount of weapons?

3. What level of runway closure can be obtained with an independently specified strategy?

Objectives

The objective of this research is to develop a computer program which will answer the above questions. Sub-objectives are:

1. Find or develop a math model to evaluate a strategy.

   a. Model must be capable to evaluate any strategy.

   b. Model must be relatively fast in terms of computer CPU time.

2. Find or develop a search algorithm to identify the "best" strategy for a given mission.

Assumptions

In the development of the search to find "best" strategies or evaluate any given strategy, the following assumptions are made:

1. The weapons of any strategy are independent, individually targeted.

2. The error distribution of any weapon is circular normal centered at the aim point with a specified CEP value.

3. Only one type of weapon will be used to close the runway for a given strategy.

4. The survivability of all weapons is one.

5. The reliability of all weapons is one.

6. There is no bomb damage assessment capability to update the targeting strategy.

7. The minimum launch window (minimum length/ minimum width needed for takeoff) must be composed of actual runway.

8. If a minimum launch window exists, it is assumed to be accessible to an aircraft for takeoff.

Scope and Limitations

This model considers only conventional weapons in destroying non-reinforced concrete runways.

General Procedure

Basically three procedures are developed in this research.

1. Starting from the independent strategy, the prob-

lem is to calculate the probability of runway closure when the strategy is completely specified. The procedure developed here relies on either a discrete approximation or a monte-carlo simulation. The discrete approximation is faster (on the computer) for problems where the effective minimum takeoff width is approximately half or more of the effective runway width. If not, then the number of calculations necessary for the discrete approximation makes it less efficient than the monte-carlo simulation, which is a constant run-time routine.

2. The next problem is the highest probability of runway closure for a given number of weapons. This problem requires a search procedure that will consider the possible combinations of weapons and aim points and aim point locations to obtain the highest probability of runway closure. The various combinations of number of weapons per aim point and aim point locations are evaluated by procedure 1 to find the strategy which yields the highest probability of runway closure.

3. The last problem requires the level of runway closure to meet or exceed a predetermined level of runway closure. The total number of weapons will be varied between a minimum and a maximum amount. Each number of weapons is individually searched by procedure 2 to obtain the highest probability and stops for the smallest number of weapons that meets or exceeds the predetermined level of runway closure. This number is reported as the "best"

number of weapons needed to obtain the predetermined level.

Organization

The first phase involves finding a computer routine that can evaluate a given strategy to answer the following question: What level of runway closure can be obtained with an independently specified strategy? Various existing modeling approaches are reviewed in Chapter III. In Chapter III, the specific computer routine will be described which was chosen, based on both speed and accuracy. A relatively fast computer routine is necessary because of its repeated use in the search algorithm.

The second phase involves selecting a search routine that will find the "best" strategy for a given number of weapons to answer the question: What is the highest level of runway closure that can be obtained with a fixed amount of weapons? The search routine, described in Chapter IV, searches among the combinations of number of weapons per aim point and aim point locations to find the strategy that gives the highest level of runway closure.

Finally, the third phase, involves writing the overall program that will find the "best" strategy to meet or exceed a predetermined level of runway closure to answer the question: How many weapons does it take to obtain a certain level of runway closure? This program, described in Chapter V, searches for the minimum amount of weapons necessary to achieve a certain level of runway closure.

II. Background

This chapter is a review of the currently available literature pertaining to probability models. Three areas of interest are discussed. The first area is the broad subject of coverage problems where the probability of destroying a fixed target is given. Next, is a review of models that evaluate a particular runway closure strategy. The third is a review of the one model found that incorporates a searching scheme with an evaluation method, resulting in a targeting strategy.

Coverage Problems

The evaluation of targeting strategy can be logically developed by first considering the impact of a single weapon on a fixed rectangular target (e.g., runway). This type of research comes under the general heading of coverage problems. One very helpful source in this area is the "Survey of Coverage Problems Associated with Point and Area Targets" by A. R. Eckler (Ref 3). Jaiswan and Sengal looked at the problem of the expected damage area for stick and triangular pattern bombing (Ref 7). All these results consider only fixed targets; however, the problem of runway closure must consider the minimum launch window (i.e., the least amount of runway surface necessary for aircraft launch as the actual "target". In other words, the objective of bomb damage is to deny an area equal to the minimum launch window anywhere on the runway. Thus, just computing the expected area damaged without regard to where the damage occurs is not sufficient for calculating the probability of runway closure.

Probability of Closure Models

The models that can explicitly compute the probability of runway closure are: AIDA, TSARINA, AHAB, RUNW and Manz. AIDA, TSARINA, AHAB and RUNW are all monte-carlo simulations.

AIDA (Airbase Damage Assessment) program (Ref 4), which is the principle model used by USAF/Studies and Analysis, is a large scale FORTRAN IV model that considers the whole airfield complex with up to 250 separate targets in the complex. Targeting flexibility includes 250 separate targets, such as parked aircraft, POL centers, hangars, etc. Each target can be assigned to a separate vulnerability class; hard targets such as hardened hangars may be one vulnerability class while aircraft in the open are in another. Up to 20 different vulnerability classes may be identified.

Each weapons delivery pass is described by weapon type and weapon release parameters (such as heading, altitude, air speed, dive angle, and stick length). Up to 10 different weapon types may be used in combination with up to 50 separate weapon delivery passes. Also considered are the probability of arriving at the target, attrition due to enemy fire, aiming accuracy in terms of REP and DEP and ballistic dispersion.

With regards to runway targets, only point-impact weapons are considered. The effective miss distance is equal to the damage radius. Up to 250 hits may be stored for each runway. Should the runway be closed, it identifies the minimum number of craters that have to be repaired before the runway can be reopened. An approximate computer plot of impact points is available upon request.

The probability of runway closure is calculated by the proportion of times the runway was closed during the simulation. The runway is considered closed if no rectangle equal to the minimum launch window exists anywhere on the runway without a crater. AIDA determines this by starting at one corner of the runway, positioning a rectangle equal to the minimum launch window on it and then observing whether or not the rectangle contains a crater. If it does contain a crater, the rectangle is then shifted five feet along the width of the runway and is again checked for craters. If an open rectangle does not exist (each rectangle contains one or more craters) when the procedure reaches the other end of the runway width, the rectangle is then shifted 250 feet along the length of the runway, positioned at one edge of the runway width, and the search continues until either an open rectangle is found (the entire runway is open) or the procedure reaches the opposite corner of the runway and no open rectangle is found (the entire runway is closed).

TSARINA (Ref 5) is a version of AIDA that makes the AIDA model compatible with the TSAR (Theater Simulation of Airbase Resources) model, thereby building a very large scale model.

AHAB (Attacking Hardened Air Bases) by RAND Corporation (Ref 10) is an interactive computer simulation designed to aid decision makers in finding an airbase attack plan which maximizes their value function. The value function must do three things:

1. Reflect the natural ordering of preferences - in fact, rank losing two of your aircraft as superior to losing three of them.

2. Express various components of an outcome in comparable terms. Is losing two of your own planes to destroy four of the enemy's better or worse than losing five to destroy eight?

3. Distinguish among uncertain outcomes. For example, although we may know that there is a fifty-fifty chance that an attacking aircraft will destroy a hangarette, we cannot know how many hangarettes will be destroyed during a particular mission, nor can we be certain of how many aircraft will be in those hangarettes. The value function places values on the probability distributions of outcomes. We should be able to decide whether a 50 percent chance of losing six aircraft and a 50 percent chance of losing none, is better than, worse than, or the same as 100 percent chance of losing three, all other things being equal.

The von Neuman-Morgenstern utility function is the value function used in this model. This function reflects not only the appropriate values or utilities of outcomes, but a characterization of the decision maker's attitudes toward risk.

The value function is linear and has four arguments: the number of enemy aircraft destroyed, the number of aircraft shelters or hangarettes destroyed, the number of hours the runway complex is closed, and the number of friendly aircraft lost in the attack.

AHAB is an interactive monte-carlo simulation written in JOSS (a RAND language). The target complex consists of open aircraft, hangarettes and runways. Only one weapon type is considered for a problem, and the attack is assumed to come across the runway perpendicular to the runway at evenly spaced "cuts" along the runway.

The calculation of the probability of runway closure is to look for a gap of 50+d feet, assuming a minimum take-off width of 50 feet, along any of the "cuts" where d is equal to two times the damage radius of the weapon. If a gap of 50+d is found along any "cut" then the runway is open, if not, then the runway is closed.

RUNW (Ref 12) is a NATO program specifically designed for the calculation of runway interdiction probability. RUNW is a monte-carlo simulation model that computes only the probability of closure for one runway. The target is a rectangular runway and the minimum launch window is a rectangle (minimum launch length by minimum launch width). An attack is described by the weapon type, attack heading (from parallel to the runway to perpendicular to the runway), whether the weapons are released individually, in salvos or in sticks, the aiming error (normally distributed), the weapon dispersion error (normally or uniformly distributed).

The probability of runway closure for RUNW is the proportion of times the runway is closed over the number of iterations used. The procedure for finding an open minimum launch window first orders the impact points on the runway along the x-axis (length) then looks at the subsequent spaces between these ordered points. If any space is greater than the minimum launch length, then the runway is open. If there are no spaces greater than the minimum launch length, it then looks at spaces with one point in between. If there is a space greater than the minimum launch length now, the lateral spacing between the runway edge and the bomb is checked against the minimum launch width, and if greater, then the runway is open. The search continues until the case when four interior points exist in the spacing. If no minimum launch window is found, then the runway is considered closed.

The model by Dr. Manz (Ref 9) calculates the probability of runway closure for cluster munitions that are uniformly distributed. This is an analytical model based on the binomial distribution of submunitions falling in a given subarea of the runway and a differential equation argument to compute the probability of runway closure.

The probability of runway closure $P(C|N_o)$, given that

N_o submunitions are aimed at the runway, is

$$P(C|N_o) = \sum_{N=0}^{N_o} P(C|N)\, P(N|N_o) \qquad (1)$$

where $P(N|N_o)$ is the probability that N submunitions will impact on the runway given that N_o submunitions were aimed at the runway. This is a binomial distribution of the submunitions with a probability P_o for any one submunition impacting the runway.

$P(C|N)$ is theprobability of runway closure given that N submunitions have impacted the runway. This probability can be evaluated for a given rectangle on the runway. Then the rectangle is moved a distance, dx, in the x direction, and dy in the y direction. The probability that a rectangle that was closed will be opened up by the move is calculated by solving a differential equation.

<u>Runway Closure Strategy Model</u>

Only one model has attempted to search for the "best" strategy in terms of aim point locations and number of weapons per aim point. The "best" strategy is defined as the strategy which will obtain the required level of runway closure with the least number of weapons. This model is by Capt. Pemberton (Ref 11) and contains two parts.

The first calculates the probability of runway closure for a given strategy by either a discrete approximation or a monte-carlo simulation. The technique used depends on the speed of the calculation. The monte-carlo simulation requires a lot of computer time to evaluate any strategy. The discrete approximation evaluation time depends on the dimensions of the runway and the MLW. When the MLW's width is from 1/3 to approximately equal to the runway's width, the discrete approximation requires fewer calculations and is faster than the monte-carlo simulation. When the MLW's width decreases in proportion to the runway's width, the number of calculations increase, and when the MLW's width is approximately equal to 1/3 of the runway's width, the number of calculations required in the discrete approximation make the calculating time greater than that required by the monte-carlo simulation. Thus, for situations where the MLW's width is less than 1/3 of the runway's width, the program uses the monte-carlo simulation because it is faster.

The second part is a search routine which finds the "best" strategy. The routine systematically searches over the combinations of aim point locations and number of weapons per aim point by first selecting the "best" aim point locations then allocating weapons to these aim points until the required level of runway closure is obtained.

Comparison of Models

From this research it was concluded that the model by Pemberton was the closest to meeting the requirements of the problem. However, the Pemberton model was considered to have weaknesses in the discrete approximation section and needed a better search routine. The objectives of this effort are to improve the Pemberton model and apply the model in a program which could be a useful decision aid to a military planner.

III. Evaluation of a Given Strategy

Expansion of the problem to throwing a point at an extended target

When considering weapons with a damage radius greater than zero, the impact point of the weapon need not coincide with the target to damage it. While the weapon may miss the target by an amount equal to the radius, it can still effectively damage the target. On the other hand, by expanding the target's boundaries by an amount equal to the damage radius all around and considering the weapon as a point weapon, damage to the target could then be determined by observing whether or not the impact point is within the extended boundaries (see Figure 2).

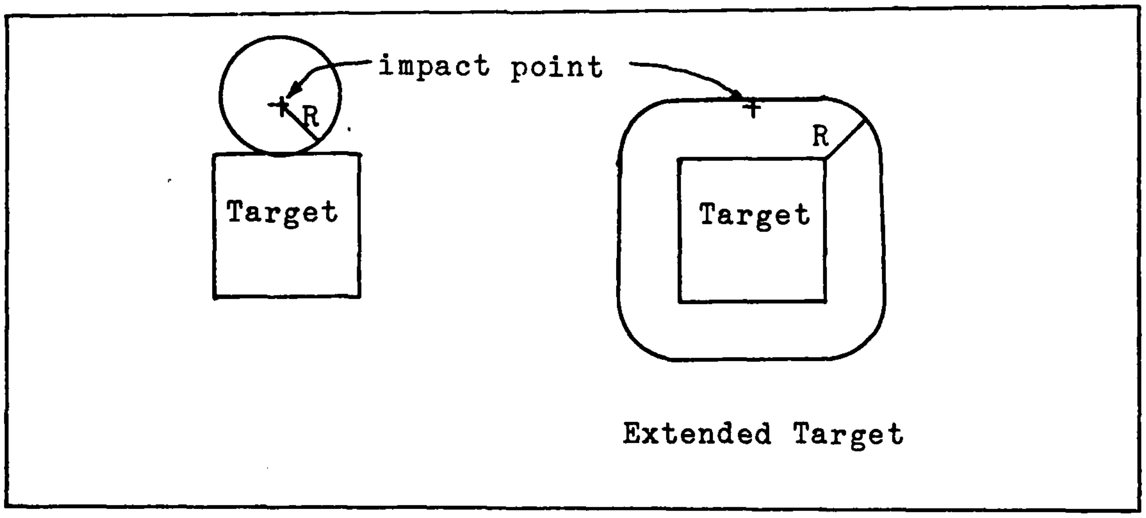


Figure 2. Extended Target Boundary

In this research, runway and minimum launch window dimensions are converted into extended dimensions for the probability calculations. For example, the extended minimum takeoff width (ETOWID) is equal to the minimum takeoff width (TOWID) plus twice the damage radius (DAMRAD).

<u>Reduction of the two dimensional problem to a one dimensional problem</u>

The ideal approach to the problem of computing runway closure probability involves computing the probability that an open space in both length and width does not exist anywhere within the runway boundaries. For an overall approach, one can order all of the weapons that impact on the runway in increasing order, from the origin, see figure 3.

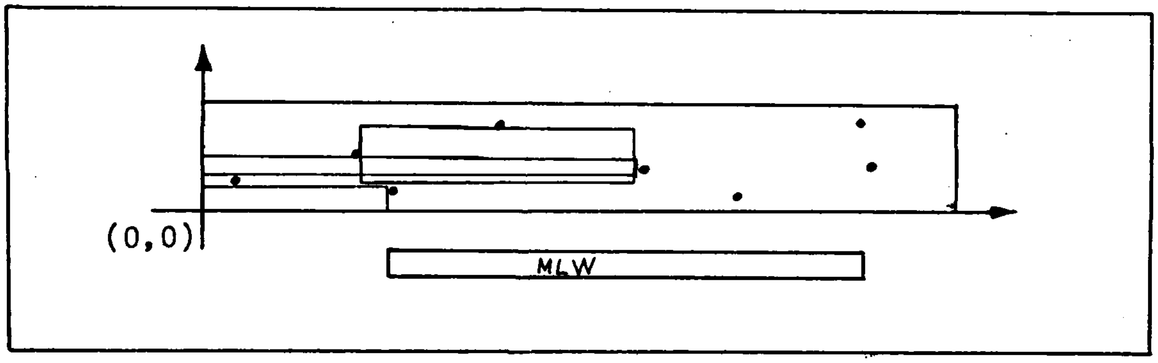


Figure 3. Ideal approach for probability of closure

Under this arrangement, rectangular spaces are formed by either four points or by the runway boundaries and impact points. These rectangles are then compared to the minimum launch window. If any rectangle is larger than the MLW, then the runway is open. The probability that any rectangle is larger than the MLW depends on the joint probability distribution of all impact points.

This two-dimensional problem can be reduced to a one dimensional problem, since runways are many times longer than they are wide. The closing of a given MLW by denying its length is dependent on closing of that MLW by denying its width. See figure 4. Runway closure can be accomplished if we have one "cut" per MLW, where a cut is defined to be a strategy that attempts to deny a space along a line perpendicular to the runway length equal to the MLW width. Then the probability that a MLW is closed is equal to the probability of cut. Conversely, the probability that a MLW is open is equal to the probability of not cutting the runway at that location. This assumes that, given a successful cut, no weapon extends beyond the length of the MLW. This can be virtually assured by aiming the cut at least three sigma (see appendix B for the relationship between CEP and sigma) away from either edge of the MLW.

Thus, the probability of closure is then the product of the probabilities of cut of the individual cuts.

Arrangement of cuts along the runway and the minimum launch lengths are shown in figure 4.

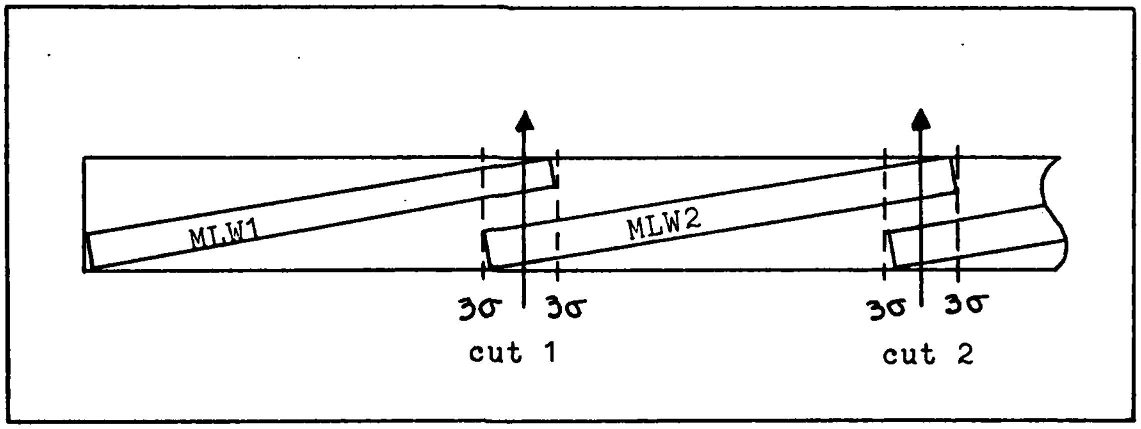


Figure 4. Individual Cut Locations

The runway is open if any cut fails to deny the minimum launch width. For example, if cut 1 fails then the length from the left edge of the runway to cut 2 is open: Since there is a gap greater than the minimum launch width in cut 1 there is at least one open minimum launch window (MLW1). If cut 2 fails, then a minimum launch window can be positioned in the space in cut 2 with sufficient length between cuts 1 and 3; thus the runway is open.

The runway is closed only if all cuts succeed. The number of cuts required is calculated by

$$NCUTS = \left[\frac{(ERUNLE - 6(STADEV))}{SRL - 6(STADEV)}\right] \qquad (2)$$

where NCUTS, the number of cuts required, is the greatest integer less than the quantity in the brackets. STADEV is the standard deviation of the impact point distribution (sigma) derived in appendix B. SRL is the shortest runway length for takeoff derived in appendix C. Thus, the probability of runway closure is

$$P_{CL} = \prod_{i=1}^{NCUTS} PC_i \qquad (3)$$

where PC_i is the probability of cut for location i. This method assumes that sigma is less than ETOLEN/6, where ETOLEN is the effective takeoff length, because of the requirement for independence of the cuts in this case.

We can now concentrate on the probability of cut (PC).

This is the probability of denying (along the width) a space equal to the minimum launch width. Three methods of calculating the probability of cut were examined: an order statistics approach, a discrete approximation approach which approximates the continuous nature of successive minimum widths by discrete steps, and monte-carlo simulation.

Order Statistics Approach

The problem is to calculate the probability that the largest space along a line perpendicular to the length of the runway is less than a certain minimum launch window's width. ξ, the largest space is the max $\{x_{(i+1)} - x_{(i)}\}$ where $x_{(i)}$ are the order statistics of the impact points from one edge of the runway, with $x_{(0)} = 0$ and $x_{(n+1)} =$ runway width.

If we assume a uniform distribution across the runway and assume that the runway width is 1, then the probability that ξ is greater than v (the minimum launch window's width) has been derived by David (Ref 1), and is given by:

$$\Pr\{\xi > v\} = n(1-v)^{n-1} - \binom{n}{2}(1-2v)^{n-1} + \cdots (-1)^{i+1}\binom{n}{i}(1-iv)^{n-1} \tag{4}$$

n = number of spaces = number of weapons + 1

where the series continues as long as (1-iv)>0. (Ref 1:81)

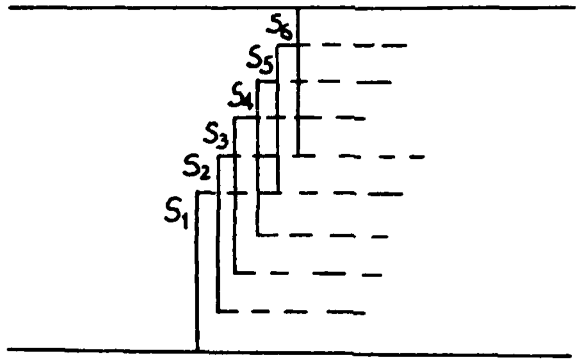
The problem in this research is to apply this procedure not to independent identically distributed uniform distributions, but to independent non-identical normally distributed bombing distributions. Unfortunately, when considering the normal distribution, the order statistics approach gets more complex. When the order statistics of the joint normal distribution are superimposed on a fixed range (runway width) this introduces two more order statistics not from the previous joint distribution. The next step is to take the difference between the order statistics begining from one end of the runway width to the other end. These differences are then ordered and we are interested in the largest ordered space statistic. The distribution of this order statistic could not be found, and derivation, though perhaps possible, was beyond the scope of this research (Ref 6).

Since an exact solution of the problem could not be found, the problem was divided into discrete subproblems discussed in the next section.

Discrete Approximation

This approach yields an analytic solution based on the set theory concept of unions and intersections of non-independent subevents to express a certain event. The event of interest is the runway cut denoted (RC). The runway is not cut (i.e., open) (denoted $\overline{RC}$) if there exists a space equal to or greater than the minimum launch window's width. The runway is thought of as containing some number of discrete overlapping minimum launch window widths, illustrated in figure 5. These overlapping minimum launch widths are called sections (s) and there are NSPW (number of sections per runway width) sections.

NSPW = 6



The s_i's are the discrete minimum launch window widths that are to be evaluated. The event $\{s_i \text{ closed}\}$ occurs when a weapon lands within section s_i. The event $\{s_i \text{ open}\}$ occurs when all weapons miss section s_i. NSPW is the number of sections s_i per runway width.

Figure 5. Relation between s_i's and runway width

The event RC is equivalent to the event $\{\text{all } s_i \text{ are closed}\}$, where the event $\{s_i \text{ closed}\}$ is when the section sustains some damage (at least one impact point is within s_i) and the event $\{s_i \text{ open}\}$ is when the section does not sustain any damage.

$$RC = \bigcap_{i=1}^{NSPW} \{s_i \text{ closed}\} \qquad (5)$$

The events $\{s_i \text{ open}\}$ are not independent so the probability of RC denoted (PC) cannot be expressed as the product of the individual probabilities of s_i. The event $\overline{RC}$ is equivalent to the event $\{\text{at least one } s_i \text{ is open}\}$.

$$\overline{RC} = \bigcup_{i=1}^{NSPW} \left\{ s_i \text{ open} \right\} \tag{6}$$

So, the probability that the runway is not cut $\overline{PC}$, ($\Pr\left\{\overline{RC}\right\}$) can be expressed by the probability multiplication law as:

$$\begin{aligned}\Pr\left\{\overline{RC}\right\} = \overline{PC} = &\sum_{i=1}^{NSPW} \Pr\left\{s_i \text{ open}\right\} \\ &- \underset{i<j}{\sum_{i=1}^{NSPW-1} \sum_{j=2}^{NSPW}} \Pr\left\{s_i, s_j \text{ open}\right\} \\ &+ \underset{i<j<k}{\sum_{i=1}^{NSPW-2} \sum_{j=2}^{NSPW-1} \sum_{k=3}^{NSPW}} \Pr\left\{s_i, s_j, s_k \text{ open}\right\} \\ &- \cdots + \cdots (-1)^{NSPW-1} \Pr\left\{s_1, s_2, \cdots, s_{NSPW} \text{ open}\right\}\end{aligned} \tag{7}$$

(Ref 8:33)

There are $2^{NSPW}-1$ terms in this expression. In Pemberton's UNION routine (Ref 11:20) all $2^{NSPW}-1$ terms are calculated. This routine can be made more efficient by observing that many of these terms cancel with each other. The event $\left\{s_i, s_j \text{ open}\right\}$ occurs when no weapon damages either the ith or the jth section. If s_i and s_j overlap ($s_i \cap s_j \neq \emptyset$) then the event $\left\{s_i, s_j \text{ open}\right\}$ occurs when all weapons land above the upper section and/or below the lower section. If, however, there exists a section s_k, such that s_k is wholly contained in $\left\{s_i \cap s_j\right\}$, then the event $\left\{s_i, s_j \text{ open}\right\}$ is equivalent to the event $\{s_i, s_j, s_k \text{ open}\}$. In like manner all events that include s_i, s_j and wholly contained sections and combinations of sections in $\{s_i \cap s_j\}$ are equivalent events and their respective probabilities are equal.

It can be shown that if s_1 and s_{NSPW} overlap ($s_1 \cap s_{NSPW} \neq \emptyset$), then for every s_i, s_j ($i \neq j \neq i+1$) there exists equal probabilities with opposite signs that cancel with each other. For example, if NSPW = 4, and ($s_1 \cap s_4 \neq \emptyset$) then:

$$\overline{PC} = \sum_{i=1}^{4} \Pr\{s_i \text{ open}\} - \sum_{\substack{i=1\\i<j}}^{3} \sum_{j=2}^{4} \Pr\{s_i, s_j \text{ open}\}$$

$$+ \sum_{\substack{i=1\\i<j<k}}^{2} \sum_{j=2}^{3} \sum_{k=3}^{4} \Pr\{s_i, s_j, s_k \text{ open}\}$$

$$- \Pr\{s_1, s_2, s_3, s_4 \text{ open}\} \tag{8}$$

$$\begin{aligned} &= \Pr\{s_1 \text{ open}\} + \Pr\{s_2 \text{ open}\} + \Pr\{s_3 \text{ open}\} + \Pr\{s_4 \text{ open}\} \\ &\quad - \Pr\{s_1, s_2 \text{ open}\} - \Pr\{s_1, s_3 \text{ open}\} - \Pr\{s_1, s_4 \text{ open}\} \\ &\quad - \Pr\{s_2, s_3 \text{ open}\} - \Pr\{s_2, s_4 \text{ open}\} - \Pr\{s_3, s_4 \text{ open}\} \\ &\quad + \Pr\{s_1, s_2, s_3 \text{ open}\} + \Pr\{s_1, s_2, s_4 \text{ open}\} \\ &\quad + \Pr\{s_1, s_3, s_4 \text{ open}\} + \Pr\{s_2, s_3, s_4 \text{ open}\} \\ &\quad - \Pr\{s_1, s_2, s_3, s_4 \text{ open}\} \end{aligned}$$

Since ($s_1 \cap s_4 \neq \emptyset$) and sections s_2 and s_3 are wholly contained in $\{s_1 \cap s_4\}$ then the events:

$$\{s_1, s_3 \text{ open}\} \doteq \{s_1, s_2, s_3 \text{ open}\}$$

$$\{s_1, s_4 \text{ open}\} \equiv \{s_1, s_2, s_4 \text{ open}\} \tag{9}$$
$$\{s_2, s_4 \text{ open}\} \equiv \{s_2, s_3, s_4 \text{ open}\}$$

are equivalent events and their probabilities are equal. Each probability term has an equal probability term with the opposite sign so they cancel with each other leaving only these terms:

$$\Pr\{s_1 \text{ open}\} + \Pr\{s_2 \text{ open}\} + \Pr\{s_3 \text{ open}\} + \Pr\{s_4 \text{ open}\}$$
$$- \Pr\{s_1, s_2 \text{ open}\} - \Pr\{s_2, s_3 \text{ open}\} - \Pr\{s_3, s_4 \text{ open}\}$$

or

$$\overline{PC} = \sum_{i=1}^{NSPW} \Pr\{s_i \text{ open}\} - \sum_{i=1}^{NSPW-1} \Pr\{s_i, s_{i+1} \text{ open}\} \tag{10}$$

The probability $\Pr\{s_i \text{ open}\}$ is equal to the joint probability that every weapon impacts outside the section s_i. For example, consider two weapons and an arbitrary section s_i illustrated in figure 6.

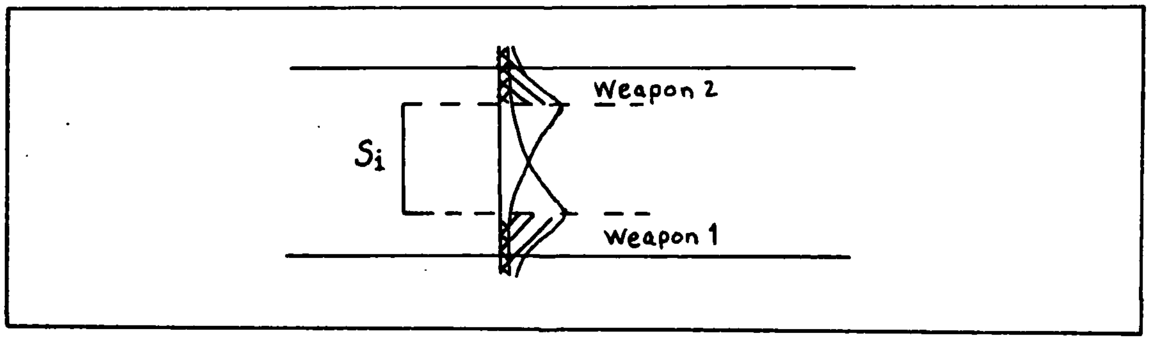


Figure 6. $\Pr\{s_i \text{ open}\}$

The probability that s_i is open from weapon 1 is the probability that weapon 1 impacts above s_i or impacts below s_i. The probability that s_i is open from weapon 2 is the probability that weapon 2 impacts above or below s_i. Since weapons 1 and 2 are independent, that is weapon 1's impact has no influence on weapon 2's impact, the probability that s_i is open is the product of these two probabilities or

$$\Pr\{s_i \text{ open}\} = \Pr\{\text{weapon 1 impacts above or below}\} \cdot \Pr\{\text{weapon 2 impacts above or below}\} \quad (11)$$

In general $\Pr\{s_i \text{ open}\} = \prod_{i=1}^{n} \Pr\{\text{weapon } i \text{ impacts above or below section } i\}$, where n represents the total number of weapons aimed at that cut. Because s_i and s_{i+1} overlap, the probability $\Pr\{s_i, s_{i+1} \text{ open}\}$ has the same form as the probability $\Pr\{s_i \text{ open}\}$ but the impacts are now above and below both s_i and s_{i+1}. This is equivalent to the probability that each weapon impacts above the upper section and impacts below the lower section. In general:

$$\Pr\{s_i, s_{i+1} \text{ open}\} = \prod_{i=1}^{n} \Pr\{\text{weapon } i \text{ impacts above } s_{i+1} \text{ or below } s_i\} \quad (12)$$

The case where i=2 is illustrated in figure 7.

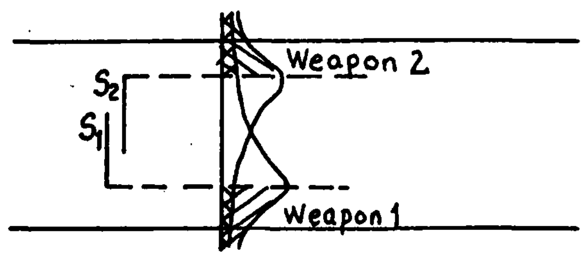


Figure 7. $\Pr\{s_i, s_{i+1} \text{ open}\}$

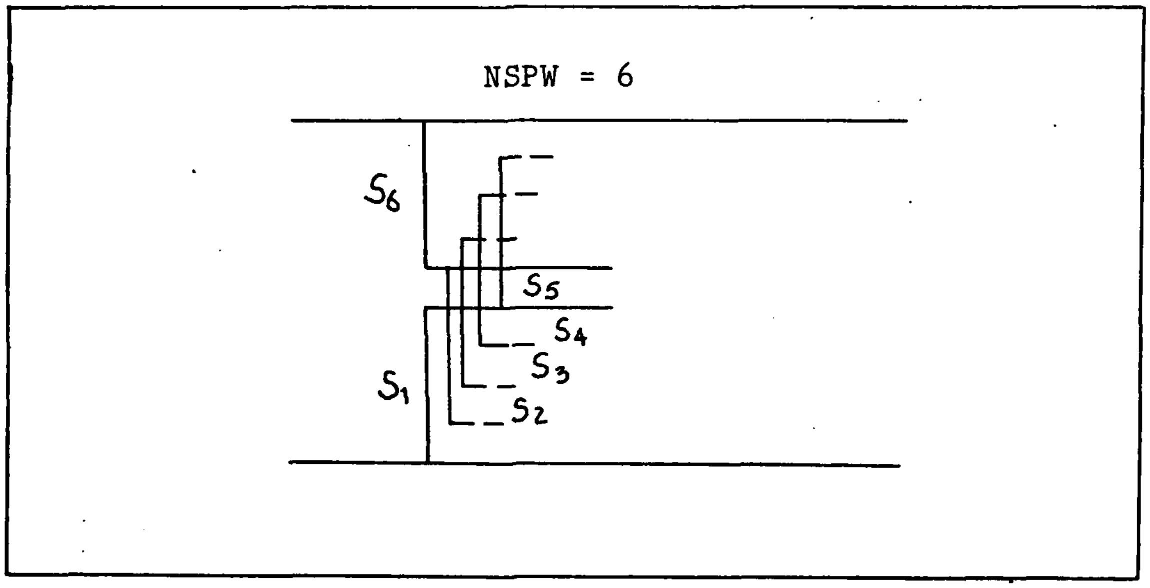


Figure 8. $s_1 \cap s_{NSPW} = \emptyset$

If s_1 and s_{NSPW} do not overlap, then some of the pairs of equivalent events identified when s_1 and s_{NSPW} overlap do not exist. These pairs have been identified for two cases.

Case I. $s_1 \cap s_{NSPW} = \emptyset$, $s_1 \cap s_{NSPW-1} \neq \emptyset$, see figure 8. From the figure we see that the event $\{s_1, s_{NSPW} \text{ open}\}$ is not equivalent to the event $\{s_1, s_2, s_{NSPW} \text{ open}\}$. Thus we need two extra terms in addition to the terms identified in equation 10.

$$\overline{PC} = \sum_{i=1}^{NSPW} \Pr\{s_i \text{ open}\} - \sum_{i=1}^{NSPW-1} \Pr\{s_i, s_{i+1} \text{ open}\} - \Pr\{s_1, s_{NSPW} \text{ open}\} + \Pr\{s_1, s_2, s_{NSPW} \text{ open}\} \tag{13}$$

where

$$\Pr\{s_1, s_{NSPW} \text{ open}\} = \sum_{i=1}^{n} \Pr\{\text{weapon i impacts above } s_{NSPW}, \text{ between } s_{NSPW} \text{ and } s_1 \text{ or below } s_1\}$$

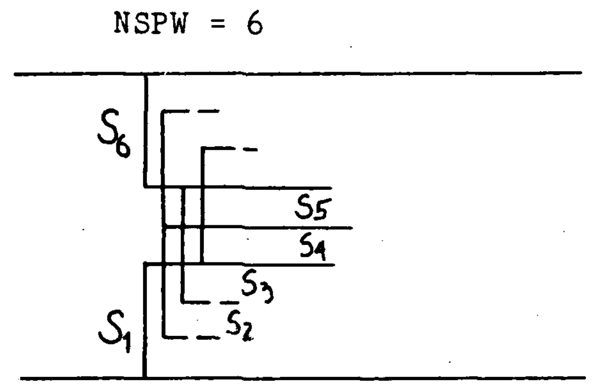


Figure 9. $s_1 \cap s_{NSPW-1} = \emptyset$

Case II. $s_1 \cap s_{NSPW} = \emptyset$, $s_1 \cap s_{NSPW-1} = \emptyset$, $s_1 \cap s_{NSPW-2} \neq \emptyset$, see figure 9. From this figure we see that the event $\{s_1, s_{NSPW}$ open$\}$ is not equivalent to the event $\{s_1, s_2, s_{NSPW}$ open$\}$. The event $\{s_1, s_{NSPW-1}$ open$\}$ is not equivalent to the event $\{s_1, s_2, s_{NSPW-1}$ open$\}$ and the event $\{s_2, s_{NSPW}$ open$\}$ is not equivalent to the event $\{s_2, s_3, s_{NSPW}$ open$\}$, and the event $\{s_1, s_{NSPW-1}, s_{NSPW}$ open$\}$ is not equivalent to the event $\{s_1, s_2, s_{NSPW-1}, s_{NSPW}$ open$\}$. Thus these eight terms need to be added to the terms identified in equation 10:

$$-\Pr\{s_1, s_{NSPW} \text{ open}\} - \Pr\{s_2, s_{NSPW} \text{ open}\} - \Pr\{s_1, s_{NSPW-1} \text{ open}\}$$
$$+\Pr\{s_1, s_2, s_{NSPW} \text{ open}\} + \Pr\{s_2, s_3, s_{NSPW} \text{ open}\}$$
$$+\Pr\{s_1, s_2, s_{NSPW-1} \text{ open}\} + \Pr\{s_1, s_2, s_{NSPW-1}, s_{NSPW} \text{ open}\}$$
$$+\Pr\{s_1, s_2, s_{NSPW-1}, s_{NSPW} \text{ open}\}$$

Further cases can be examined but the terms needed increase as $2^{m+1}-2$ where m stands for the largest m such that $s_1 \cap s_{NSPW-m} \neq \emptyset$ is true, for example, if the largest non-overlapping sections are s_1 and s_{NSPW-3} then m=3, and the number of extra terms to be added to the terms in equation 10 is:

$$\sum_{i=1}^{m} 2^{i+1}-2 = 2 + 6 + 14 = 22 \text{ extra terms.}$$

If the STEP size, which is the distance between the lower ends of two adjacent sections is equal to five feet then with the following ranges of input variables:

RUNWID	-	(100, 150)
TOWID	-	(50, 150)
W	-	(250, 2000)

The worst case is where sections s_1 and s_{NSPW-1} do not overlap which are the conditions for case II discussed above. Thus using this modified discrete approximation routine the probability of cut can be calculated with a maximum of 2(NSPW)-1+2+6+14 = 2(NSPW)-1+22 = 2(NSPW)+21 terms. The number of terms necessary for Pemberton's UNION routine requires calculating all $2^{NSPW}-1$ terms. This represents a minimum savings of $2^{NSPW}-1-2(NSPW)-21 = 2^{NSPW}-2(NSPW)-22$ terms that need not be calculated. The maximum savings is $2^{NSPW}-1-2(NSPW)+1$ or $2^{NSPW}-2(NSPW)$. For example, if NSPW = 12, the new approximation routine calculates 2(12)+21 or 45 terms and Pemberton's routine calculated $2^{12}-1$ or 4095 terms. The savings for this one calculation of the probability of cut is 4095-45 or 4050 terms that need not be calculated.

Problems with input variables that exceed the above ranges and have non-overlapping sections s_1 and s_{NSPW-2} are approximated with a monte-carlo simulation.

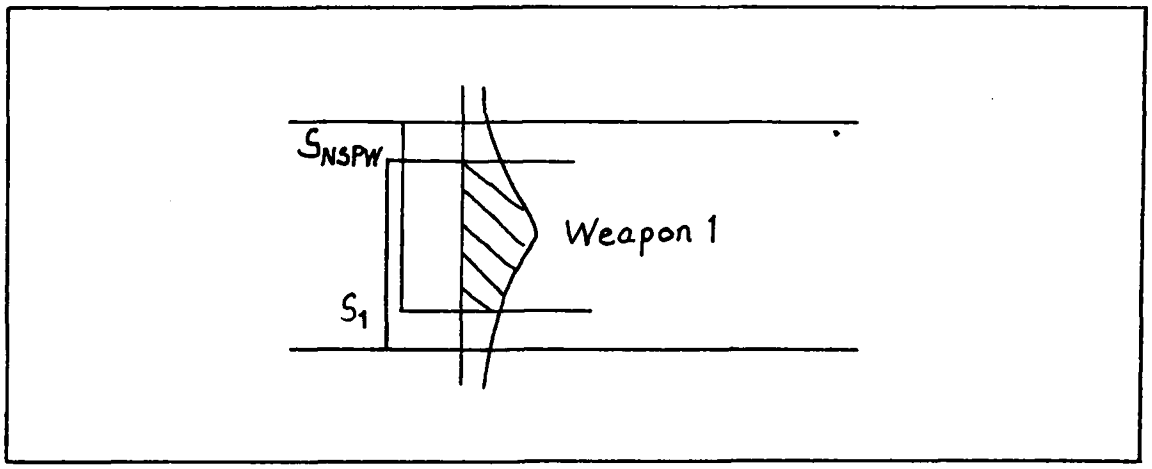


Figure 10. PC for one weapon

A special case of the discrete approximation is when considering the probability of cut when only one weapon is used. If s_1 and s_{NSPW} do not overlap, then the probability of cut is zero, since the one weapon can only damage one of the sections leaving the other section undamaged. The probability of cut when s_1 and s_{NSPW} do overlap is the probability that the weapon impacts in the intersection of s_1 and s_{NSPW}. This probability is just the area under the normal error distribution of the weapon for the intersection of sections s_1 and s_{NSPW} as shown in figure 10. The weapon must impact in the shaded area in order to cut the runway, thus the shaded area under the curve is the probability that the weapon will cut the runway.

Monte-Carlo simulation

The monte-carlo simulation is a straight forward approximation of the real world probability by "attacking" the runway many times and taking the proportion of times the runway was cut. Again, runway closure is the product of the independent probabilities of cut:

$$P_{CL} = \prod_{i=1}^{NCUT} PC_i \qquad (14)$$

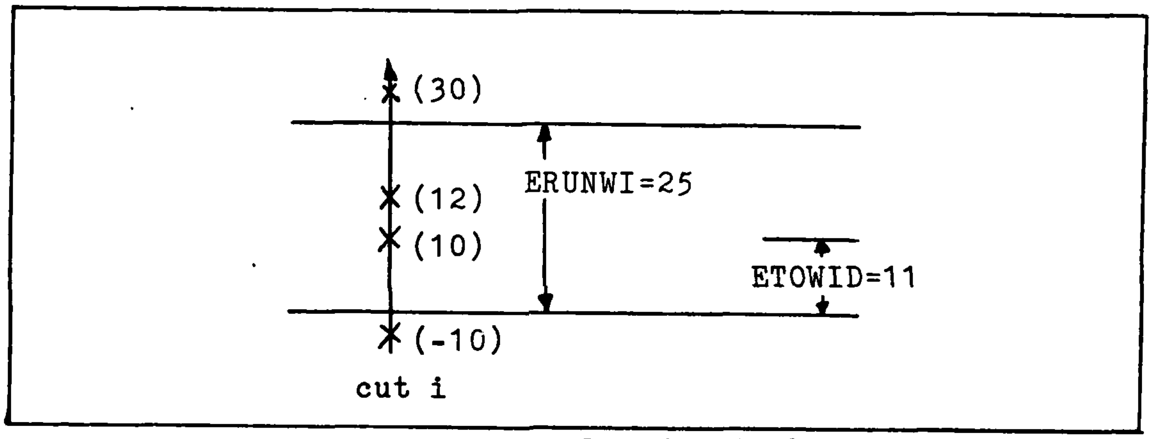


Figure 11. Simulated Attack

The runway is represented by the extended runway width (ERUNWI) and the MLW is represented by the extended takeoff width (ETOWID). Runway cut is simulated by generating weapon impact locations (hits) from a normal random number generator. This number is then translated by multiplying it by the standard deviation and then adding the mean value for that weapon distribution (STADEV and aim point for that weapon) to obtain the hit locations. After all weapons have hit, see figure 11, weapons that hit below or above the effective runway are ignored. If none of the weapons hit the effective runway, then the runway is not cut. If there is at least one weapon hit on the effective runway, the distance between the lower edge of the effective runway and the nearest hit location is compared to the effective takeoff width, if the distance is greater then the runway is not cut. Then the distance between the upper edge of the effective runway and the nearest hit location is compared to the effective takeoff width. If only one weapon hit the effective runway then the evaluation stops here. If there are more than one weapon hits on the effective runway, then the distances between two adjacent hit locations are compared against the effective takeoff width. If any of the distances is greater than the effective takeoff width, then the runway is not cut. If all the distances are less than the effective takeoff width then the runway is cut for this one "attack". The "attack" is repeated many times to obtain an estimate of the probability of cut. This results in a binomial random variable (i.e., runway is cut or not cut) and for a large number of iterations (attacks" the proportion of times the runway is cut can be approximated by a normal distribution to get an estimate on the number of iterations necessary to achieve a desired accuracy in the estimation of the probability of cut. To obtain a reasonably accurate PC estimate using the simulation, the number of iterations (NITERA) necessary is estimated as follows:

Let p equal the true probability of cut. If NITERA equals the number of trials and Θ equals the number of successful cuts observed, Bernoulli's theorem says that the difference, d, between the proportion of successes in NITERA trials and the true probability of success in a single trial tends to zero as NITERA approaches infinity. Another way of saying this is the relation:

$$\left| \frac{\Theta}{\text{NITERA}} - p \right| \leq d \qquad (15)$$

as NITERA $\rightarrow \infty$, $d \rightarrow 0$.

We wish to determine an estimate of the true probability of success such that

$$\Pr\left\{ \left| \frac{\Theta}{\text{NITERA}} - p \right| \leq d \right\} = 1-\alpha \qquad (16)$$

where Θ/NITERA is our estimate of p and $1-\alpha$ is the probability that our estimate does not deviate from p by more than d. If NITERA is large enough, then the binomial distribution can be approximated by a normal distribution. Using this approximation we can show that:

$$\text{NITERA} = \frac{z^2_{\alpha/2}\,(p)(q)}{(d)^2} \qquad (17)$$

where $z_{\alpha/2}$ is the two-tailed standardized normal statistic for the probability we seek. (Ref 13: 191-2)

In order to calculate NITERA we need an estimate of p and q. If we use the required probability of cut as the estimate of p and one minus that as the estimate of q then:

For a 99% confidence that the true probability of cut, p, lies within $\pm$ .01 of the calculated probability PC. we have

$p = .95635$

$q = .04365$

$d = .01$

$z_{\alpha/2} = 2.58$

and so

$$\text{NITERA} = \frac{(2.58)^2(.95635)(.04365)}{(.01)^2} = 2{,}778 \qquad (18)$$

With this large number of iterations necessary each time the simulation subroutine calculates a probability, the searching process consequently becomes very time consuming.

Choice of Evaluation Routine

There are two evaluation routines in the model. The first is a discrete approximation routine which is very efficient for the range of input variables identified earlier. The second routine is the monte-carlo simulation. The discrete approximation is used for problems where the difference between the effective runway width (ERUNWI) and twice the effective takeoff width (ETOWID) is less than twice the STEP size. For problems that do not meet this criterion, the monte-carlo simulation is used. This routine is not preferred because of the amount of computer time required. Because of the high probabilities of cut that are estimated, around 0.956, the number of iterations necessary to be 99% confident that the actual probability of cut lies within $\pm$ .01 of the estimated value requires approximately 2,800 iterations. The concern for speed will by an important factor in the next chapter, where the search routine evaluates many different strategies in its search for the "best" strategy.

IV. Search for the "Best" Strategy

The first step in the search is to define the minimum probability of cut for each cut that will assure an overall probability of closure that meets or exceeds the required probability of closure. This is done by taking the required probability of closure and taking the (NCUTS)th root, where NCUTS represents the number of cuts necessary to close the runway. NCUTS is the number of shortest runway lengths (SRL) in the runway with a six sigma overlap. (Refer to chapter III, figure 4) The derivation of the SRL is given in appendix C. The overlap provides a three sigma distance from the end of the SRL for each cut, in order to have independence. This minimum value for each cut is designated PCSTAR and is the minimum requirement for the probability of cut (PC) for each cut used in the search algorithm. The algorithm searches for the "best" strategy, in terms of aim points, locations of aim points and number of weapons per aim point, which yields the highest PC for a single cut. Results for the entire runway are calculated using all the cuts required where the results of each cut are independent of each other. For example, the probability of closure for NCUTS identical cuts is

$$P_{CL} = (PC)^{NCUTS} \tag{19}$$

and the number of weapons necessary for this strategy is

$$N = (n)(NCUTS) \tag{20}$$

where P_{CL} is the probability of runway closure using identical cuts. NCUTS represents the number of cuts required. N represents the total number of weapons to achieve the P_{CL} calculated above and n represents the number of weapons for the "best" strategy for each cut.

The problem is to find the "best" strategy i.e., the one which uses the least number of weapons to achieve a probability of cut at least as great as PCSTAR.

A strategy is defined by three variables: NUMAIM, NUMBOM(I), AIMLOC(I) where

NUMAIM - specifies the number of aim points under consideration.

NUMBOM(I) - specifies the number of weapons per aim point for all NUMAIM aim points.

AIMLOC(I) - specifies the location of each aim point for all NUMAIM aim points, measured in feet from one edge of the runway.

The first step is to determine a lower bound and an upper bound on the number of weapons required. The minimum number (MIN) is found by calculating the number of weapons needed if there were no variation in weapon impact points (CEP=0). This is done by counting the number of non-overlapping minimum takeoff width sections within the runway width. MIN is equal to the greatest integer not exceeding ERUNWI/ETOWID. MIN represents a minimum number of weapons necessary to cut the runway. When the actual CEP is considered, if MIN is feasible (PC greater than or equal to PCSTAR), then the least number of weapons is MIN; if not, the minimum feasible soultion will not be less than MIN.

A maximum number of weapons necessary to cut the runway (MAX) is a feasible solution (achieves a PC greater than or equal to PCSTAR.) From the calculation of the MIN number of weapons we also have the minimum number of aim points necessary to cut the runway. These MIN number of aim points are evenly spaced across the runway at locations ERUNWI/(MIN+1). This number of aim points and the aim point locations are fixed in determining MAX. The number of weapons per aim point will be the only factor varied to achieve a feasible solution. This will be done by first allocating one weapon per aim point, evaluating the strategy, allocating two weapons per aim point, evaluating the strategy, and so forth until the PC is greater than or equal to PCSTAR. The desired minimum feasible number of weapons will lie between MIN and MAX.

The second step is to find the highest PC for each number of weapons, between MIN and MAX, and to stop when the PC calculated equals or exceeds PCSTAR, as shown in figure 12, where each point represents the highest PC for a given number of weapons. With the number of weapons constant, there are many combinations of NUMAIM, NUMBOM(I), and AIMLOC(I) that are possible. Because of the symmetric nature of the problem, several assumptions will be made in order to reduce the number of combinations to be considered.

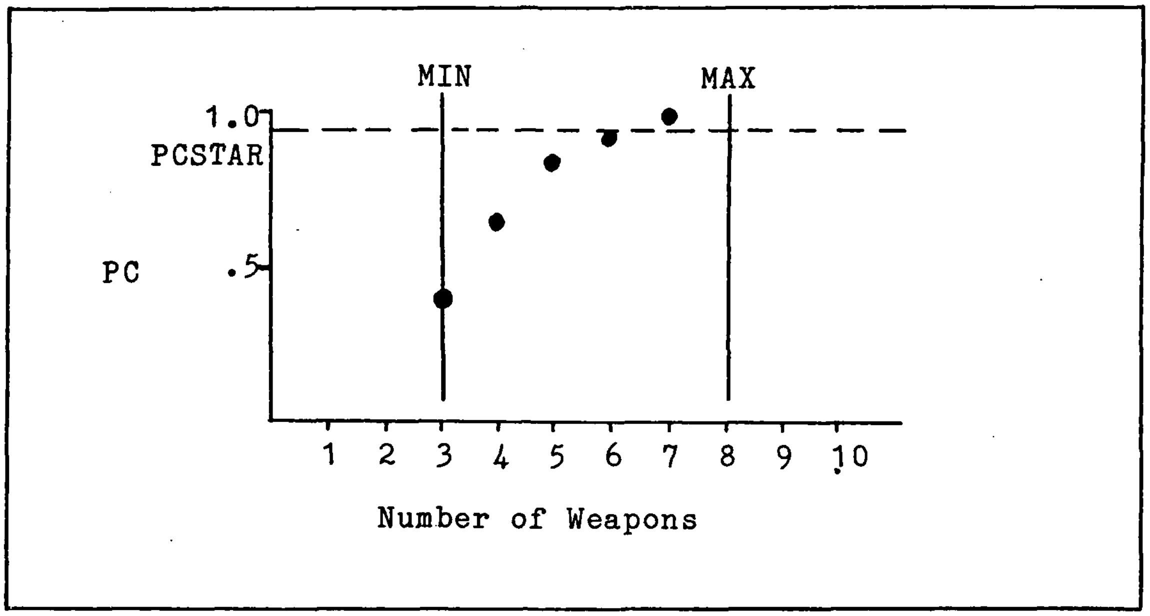


Figure 12. PC versus Number of Weapons

Assumption 1 - Symmetric numbers of weapons about the runway center are better than asymmetric numbers of weapons.

Assumption 2 - Symmetric locations of aim points about the runway center are better than asymmetric locations. Using these assumptions, it is possible to assign weapons to aim points in a systematic manner. Given the number of weapons and the number of aim points, these combinations can be easily identified, and are listed for one to nine weapons in table I. For example, if the number of aim points is one, then all the weapons will be aimed at the center. If the number of aim points is two then the number of weapons will be evenly divided (if possible) between the two aim points. For some cases there is more than one combination possible for a given number of weapons and number of aim points. For example, six weapons to be distributed among three aim points can be combined either (1,4,1) or (2,2,2). Each of these combinations is readily

Table I. Combinations of Weapons and Aim Points

		Number of Aim Points				
		1	2	3	4	5
Number of Weapons	1	(1)				
	2	(2)	(1,1)			
	3	(3)		(1,1,1)		
	4	(4)	(2,2)	(1,3,3)	(1,1,1,1)	
	5	(5)		(1,3,1)		(1,1,1,1,1)
	6	(6)	(3,3)	(1,4,1) (2,2,2)	(1,2,2,1) (2,1,1,2)	(1,1,2,1,1)
	7	(7)		(1,5,1) (2,3,2) (3,1,3)		(1,1,3,1,1) (1,2,1,2,1) (2,1,1,1,2)
	8	(8)	(4,4)	(1,6,1) (2,4,2) (3,2,3)	(1,3,3,1) (2,2,2,2) (3,1,1,3)	(1,1,4,1,1) (1,2,2,2,1) (2,1,2,1,2)
	9	(9)		(1,7,1) (2,5,2) (3,3,3) (4,1,4)		(1,1,5,1,1) (1,2,3,2,1) (1,3,1,3,1) (2,1,3,1,2) (2,2,1,2,2) (3,1,1,1,3)

identified and is a possible combination for consideration. With the NUMAIM and the NUMBOM(I) chosen, what are the "best" locations for these aim points? The aim points are located symmetrically with respect to the runway center, with the odd aim point (if there is one) being located in the center of the runway.

Because of symmetry, we only need to specify half of

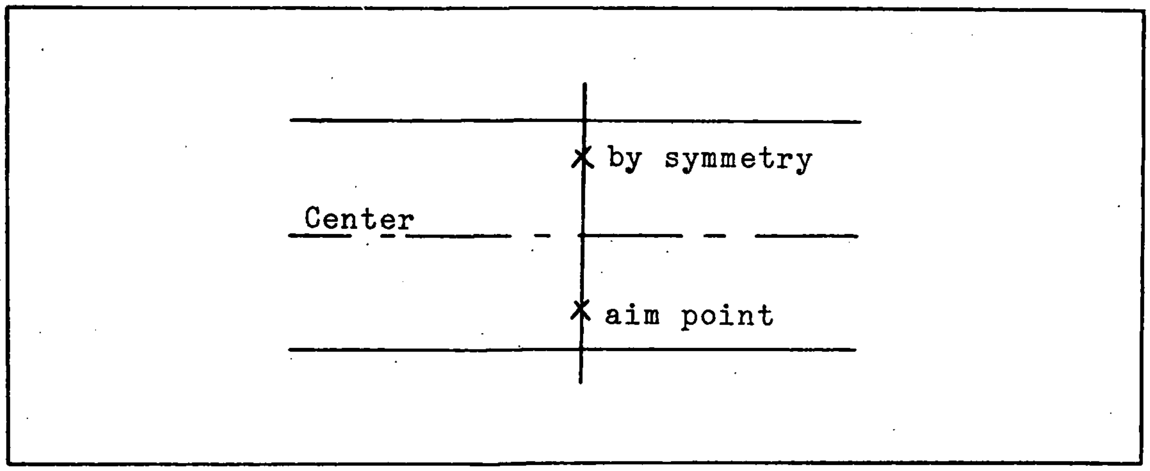


Figure 13. Location of Symmetric Aim Points

the aim point locations, with the other half being equal to the runway width minus the aim point location of the corresponding symmetric aim point. For example, if there are two aim points, aim point one is located 50 feet from the edge of the runway, then aim point two is located (RUNWID-50) feet from the edge of the runway. Thus, the two aim points can be thought of as being one aim point pair. See figure 13. The search is reduced to half of the aim point locations on half of the runway.

The response curve, probability of cut versus distance from the runway edges for a pair of symmetric aim points, was shown to be unimodal by Pemberton (Ref 11:31). This means that PC is a non-decreasing function up to the "best" location, non-increasing past the "best" location and there is only one "best" location for each pair of aim points when all other aim points are fixed. See figure 14.

The search for two "best" locations starts with aim point one at zero (one edge of the runway), aim point two,

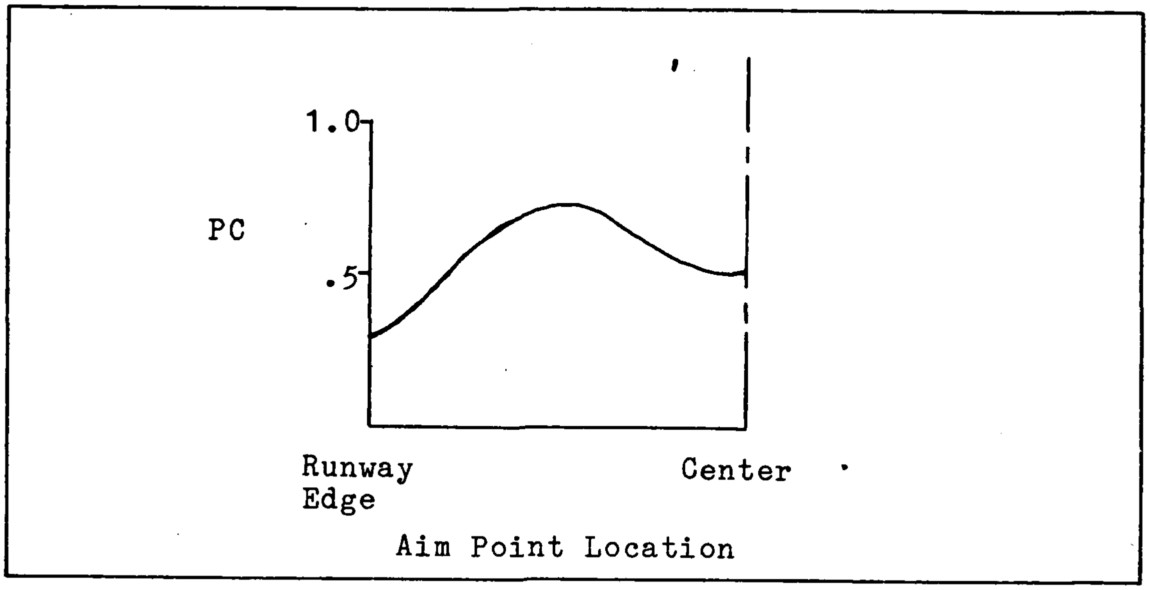


Figure 14. Response Curve for One Pair of Aim Points

by symmetry, at the other edge of the runway and evaluating the PC for this strategy. Next, the aim points are moved five feet toward the center of the runway, evaluated, and if this PC is higher than the previous PC the movement continues until, either the PC decreases, indicating we have passed the "best" location, or the aim points converge in the center of the runway, indicating one "best" location. When the PC decreases, the aim points are repositioned back to the second to the last aim point locations, and the search is repeated using one foot movements toward the center of the runway until the PC again decreases. When this occurs again, the aim point locations which gave the highest PC value is saved as the "best" locations for two aim points. The search for three "best" aim point locations is identical to the search for two "best" locations, with the middle aim point fixed in the center of the runway.

The search for four "best" aim point locations starts

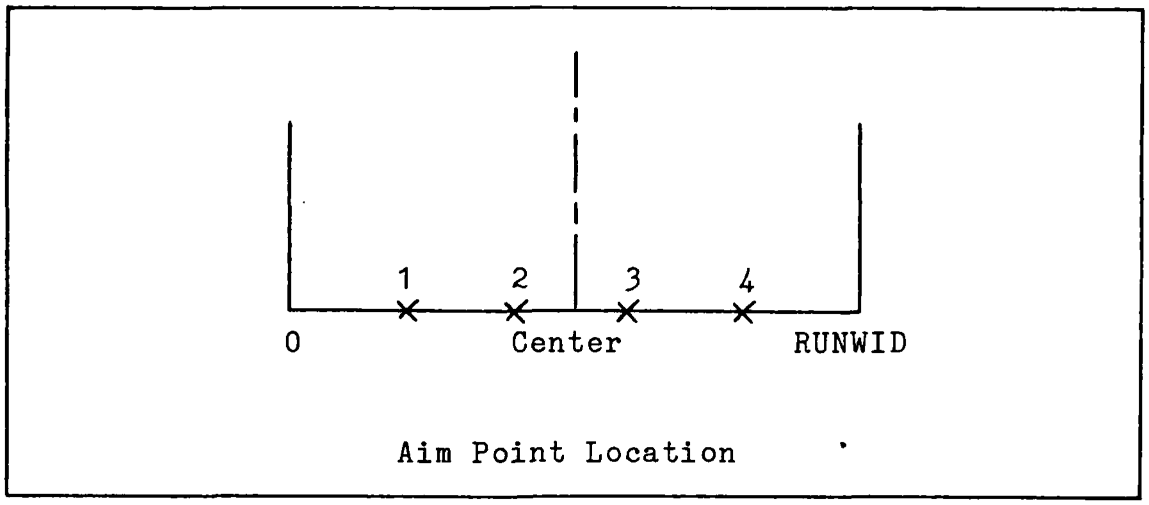


Figure 15. Initial Aim Point Locations for 4 Aim Points

with the four aim points evenly spaced across the runway width. See figure 15. Aim points 1 and 4 will be referred to as the outer pair, and aim points 2 and 3 will be the inner pair. The search uses the unimodal idea for a pair of aim points by fixing one pair and finding the "best" locations for the other pair. Initially, the outer pair are fixed and the "best" locations for the inner pair are found. Next, the inner pair are fixed at the "best" locations and the "best" locations for the outer pair of aim points are found. Next, the outer pair is fixed at its "best" locations and new "best" locations are found for the inner pair. This back and forth procedure stops when the "best" locations for both pairs of aim points do not change more than one foot. This allows the aim points to "home in" on the "best" locations. The search for five "best" locations is identical to the search for four "best" locations, with the middle aim point fixed in the center of the runway. A flow chart of the search routine is presented in figure 16.

The search begins with MIN weapons, finds the "best" number of aim points, "best" locations for these aim points and the highest PC for the "best" strategy. The "best" number of aim points is the smallest number of aim points that contribute to increase PC. If the PC with one aim point is higher than the highest PC with two aim points then the search stops for that number of weapons. If the highest PC with two aim points is higher then it is compared to the highest PC with three aim points, etc. The "best" locations for pairs of aim points are found as before. Once the highest PC has been found for a given number of weapons, the PC is compared to PCSTAR. If PC is less than PCSTAR, the number of weapons is increased by one and the search begins again. If PC is greater than or equal to PCSTAR, then the "best" strategy has been found and the search stops.

The next chapter gives an overall description of the computer program.

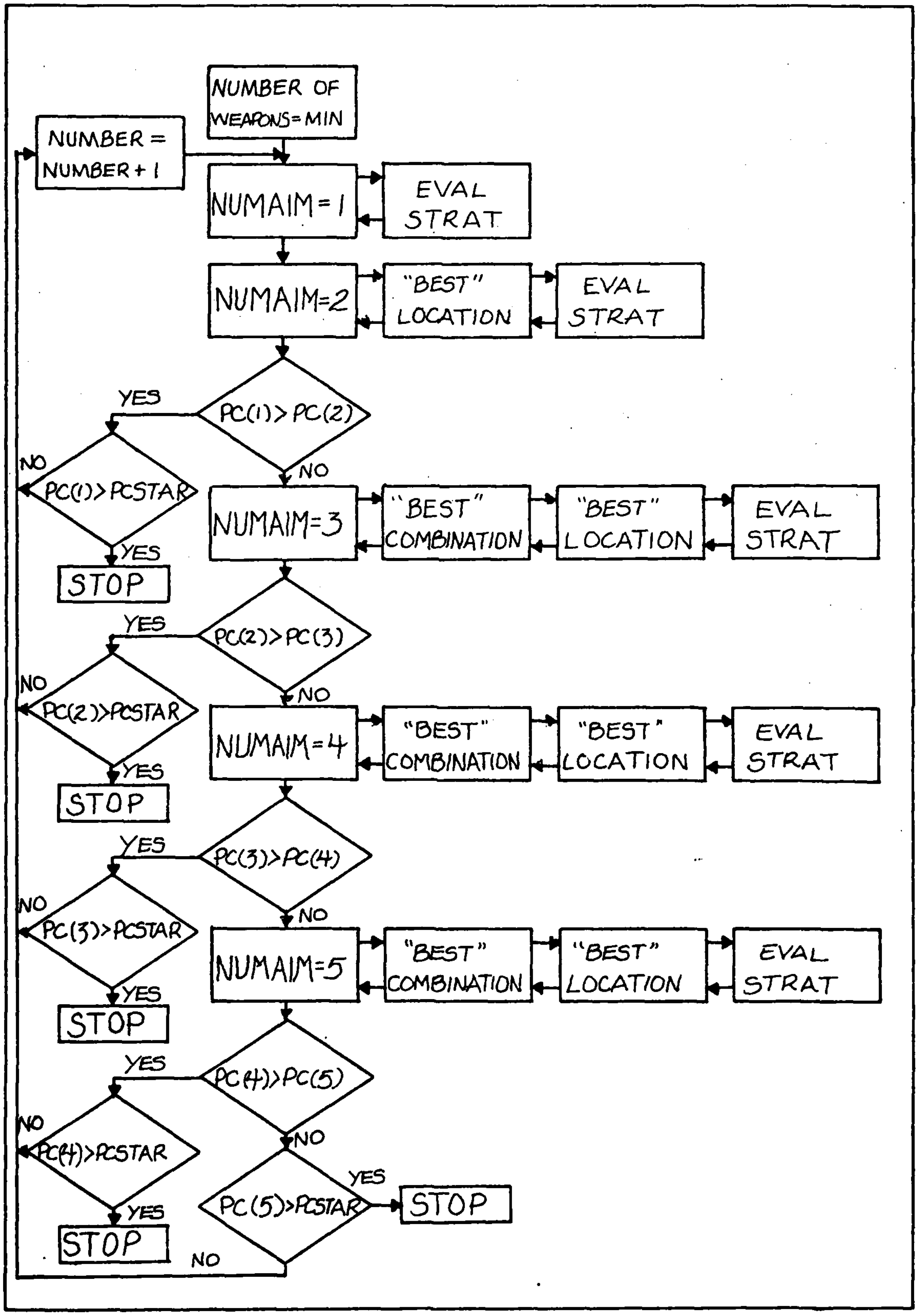


Figure 16. Search Algorithm Flowchart

V. Program Description

A computer program called RAM (Runway Attack Model) was developed to aid in planning a runway attack. The diagram in figure 17 shows the overall structure of RAM. Three separate modes of operation are available in the program. Inputs common to all three modes are:

Runway dimensions

- runway length (RUNLEN) in feet
- runway width (RUNWID) in feet

Minimum launch window (MLW)

- MLW length (TOLEN) in feet
- MLW width (TOWID) in feet

Weapon characteristics

- yield (W) in pounds TNT
- accuracy (CEP) in feet

where the input units of W in pounds TNT is converted to the weapon damage radius by the relation:

$$R = 3.54(W)^{1/3} \qquad (21)$$

(Ref 11: 58)

and CEP in feet is converted to sigma as derived in appendix B.

Mode 1. In this mode the user specifies the desired probability of runway closure (P_{CL}, $0 \leq P_{CL} \leq 1$). The program then searches for the "best" strategy, with identical strategies per cut, that will give an overall probability of runway closure equal to or greater than the desired

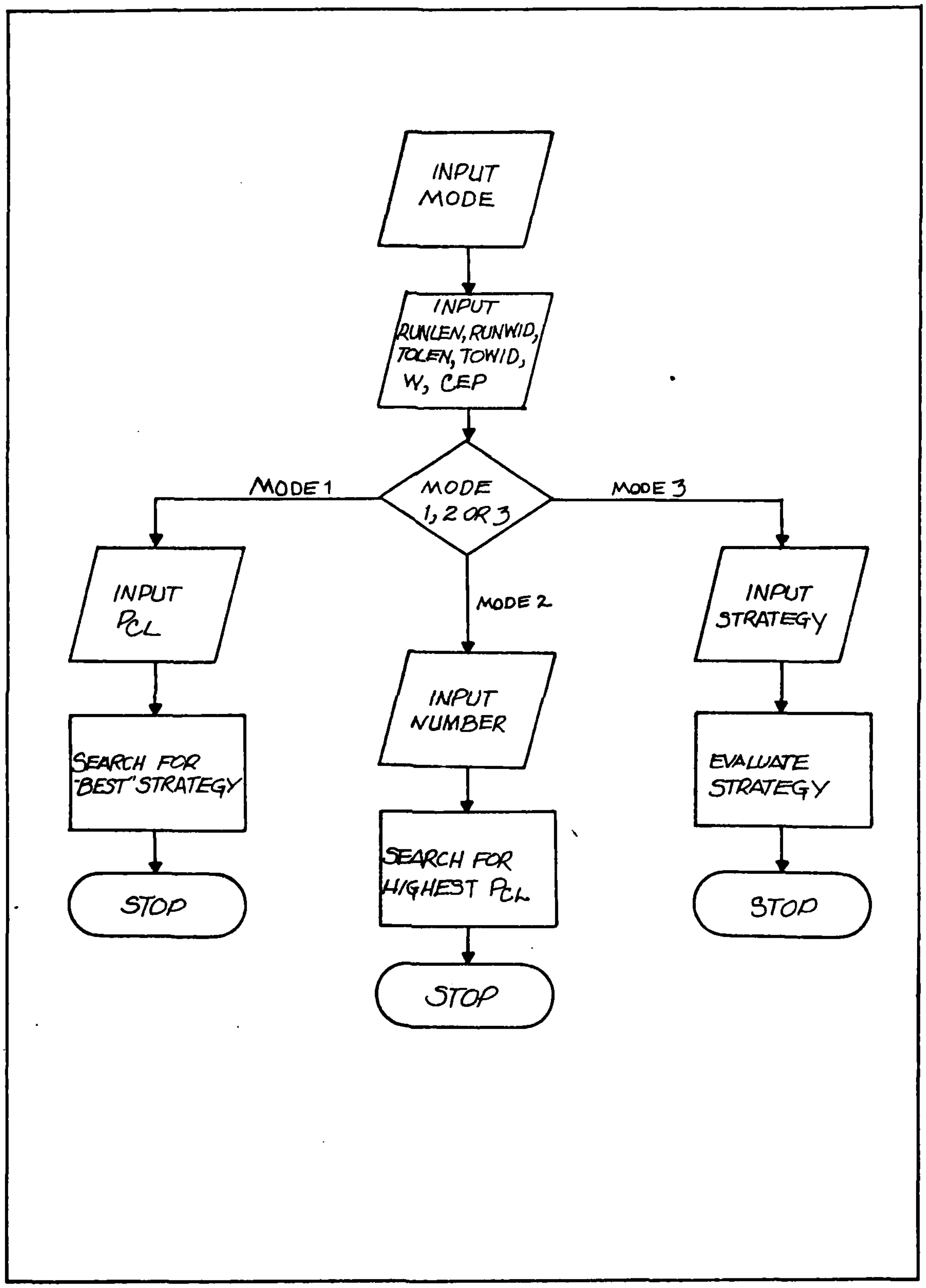


Figure 17. RAM Flowchart

probability.

Mode 2. In this mode the user specifies the number of weapons available (NUMBER). The program then searches for the strategy that gives the highest probability of runway closure.

Mode 3. In this mode the user specifies an independent strategy in terms of:

Number of aim points for each cut

Number of weapons for each aim point

Location of each aim point

with the number and location of cuts the same as that identified in mode 1. The program then uses the appropriate evaluation routine and gives the approximate probability of closure for the independent strategy. This mode does not use the search routine.

As an example of this model, the following were used as input:

Runway	
- length	8,000 feet
- width	150 feet
Minimum launch window	
- length	2,000 feet
- width	50 feet
Weapon characteristics	
- yield	250 lbs TNT
- CEP	20 feet

This problem required a search for a strategy to deny a space equal to 2,000 by 50 feet in a runway of 8,000 by 150 feet. Each weapon had a damage radius of 22.3 feet and an accuracy of 20 feet CEP. For mode 1, the desired P_{CL} was input as 0.8. The resulting "best" strategy was to cut the runway at four locations, 1750, 3250, 4750 and 6250 feet from one end of the runway. For each cut, three aim points were identified at 36, 75 and 114 feet from one edge of the runway. The number of weapons for each aim point was one. With this "best" strategy the level of runway closure was 0.91, and the total number of weapons necessary was 12. Refer to figure 18 for sample output.

In mode 2, the number of weapons available was input as 15. Because of the same input conditions, the number of cuts will not change for this example. The number of available weapons were equally divided among the cuts with the remaining weapons allocated one to a cut starting at one end of the runway. For this example, three cuts had four weapons and one cut had three weapons. The program then determines the "best" strategy to employ for the number of weapons allocated to each cut. In this case the "best" strategy, for cuts with four weapons, was to have two weapons each on two aim points located at 44 and 106 feet from one edge of the runway. For the one cut with three weapons the same strategy as found in mode 1 is the "best" where one weapon was on each of three aim points located at 36, 75 and 114 feet from one edge of the runway. The highest probability of closure with 15 weapons available

```
   RUNWAY       MINIMUM LAUNCH    WEAPON     PROBABILITY
                    WINDOW                        OF
LENGTH WIDTH  LENGTH WIDTH     YIELD     CEP   CLOSURE

 8000.  150.   2000.   50.      250.     20.     .80

RUNWAY                     8000. BY      150. FEET

MIN LAUNCH WINDOW          2000. BY       50. FEET

WEAPON CHARACTERISTICS

     YIELD   250.00 POUNDS

     CEP       20.   FEET

PROBABILITY OF CLOSURE   .91

TOTAL NUMBER OF WEAPONS   12

AIM POINTS         NUMBER OF

(LENGTH, WIDTH)   WEAPONS

1750.00    36.00       1
1750.00    75.00       1
1750.00   114.00       1
3250.00    36.00       1
3250.00    75.00       1
3250.00   114.00       1
4750.00    36.00       1
4750.00    75.00       1
4750.00   114.00       1
6250.00    36.00       1
6250.00    75.00       1
6250.00   114.00       1
```

Figure 18. Mode 1 output

was 0.96. Refer to Figure 19 for sample output.

In mode 3, the user can obtain an evaluation of any independently arrived at targeting strategy. The user may want to compare his strategy with the "best" strategy found in either mode 1 or mode 2. The user must use the same number of cuts and the same cut locations but is free to specify the total number of weapons, the number of aim points per cut, the number of weapons per aim point and the aim point locations. The independent strategy for this example was:

4 Cuts (as identified in mode 1)

- Cut 1 at 1750 feet with 2 aim points
  - aim point 1 at 50 feet with 1 weapon
- Cut 2 at 3250 feet with 3 aim points
  - aim point 1 at 25 feet with 1 weapon
  - aim point 2 at 75 feet with 2 weapons
  - aim point 3 at 125 feet with 1 weapon
- Cut 3 at 4750 feet with 2 aim points
  - aim point 1 at 50 feet with 2 weapons
  - aim point 2 at 100 feet with 2 weapons
- Cut 4 at 6250 feet with 2 aim points
  - aim point 1 at 50 feet with 1 weapon.
  - aim point 2 at 100 feet with 1 weapon.

The program evaluates the independent strategy with either the discrete approximation or the monte-carlo simulation, depending on the geometry of the problem as discussed in Chapter III. The calculated probability of runway closure

```
   RUNWAY      MINIMUM LAUNCH    WEAPON        NUMBER
                   WINDOW                        OF
LENGTH WIDTH  LENGTH WIDTH     YIELD     CEP  WEAPONS

 8000.  150.    2000.   50.      250.     20.      15


RUNWAY                        8000. BY      150. FEET

MIN LAUNCH WINDOW            2000. BY       50. FEET

WEAPON CHARACTERISTICS

     YIELD    250.00 POUNDS

     CEP        20.    FEET

PROBABILITY OF CLOSURE    .96

TOTAL NUMBER OF WEAPONS    15

AIM POINTS          NUMBER OF

(LENGTH, WIDTH)   WEAPONS

1750.00    44.00       2
1750.00   106.00       2
3250.00    44.00       2
3250.00   106.00       2
4750.00    44.00       2
4750.00   106.00       2
6250.00    36.00       1
6250.00    75.00       1
6250.00   114.00       1
```

Figure 19. Mode 2 output

was 0.68. Refer to figure 20 for sample output. Although the independent strategy had the same amount of weapons, 12, the differences in strategy produced a 0.23 difference in the expected probability of runway closure.

Thus, with these three modes, a planner can initially select the desired level of runway closure. If the number of weapons required to obtain this level is considered too large, then what level of runway closure can be obtained with a fixed amount of weapons? Or, if the "best" strategy is not possible for a particular runway attack problem, due to airfield defenses or terrain, what level of runway closure can be obtained with a modified strategy? It is hoped that in answering these three questions, a planner can select the "best" strategy that meets the requirements of a particular problem. For a more detailed guide to this program, refer to the user's guide in appendix A.

The next chapter discusses the verification and validation of the model and computer program.

```
   RUNWAY      MINIMUM LAUNCH   WEAPON
                  WINDOW
LENGTH WIDTH  LENGTH WIDTH     YIELD     CEP

 8000.  150.   2000.   50.      250.     20.

INDEPENDENT ANALYSIS

INDEPENDENT STRATEGY:

CUT        AIM POINT   NUMBER PER
LOCATION   LOCATION    AIM POINT

   1750.
                50.00         1
               100.00         1

   3250.
                25.00         1
                75.00         2
               100.00         1

   4750.
                50.00         2
               100.00         2

   6250.
                50.00         1
               100.00         1

PROBABILITY OF RUNWAY CLOSURE IS  .68
```

Figure 20. Mode 3 output

VI. Verification and Validation

Verification and validation are two related processes which increase confidence in the model. Verification is the process of making sure that the model does what the analyst intends for it to do and validation is the process of making sure that the model represents the real world. (Ref 13:208)

The verification/validation process for the strategy evaluation phase used in this research was to develop a reliable reference model (monte-carlo simulation) and compare the new model (discrete approximation) to the simulation. The validation process for the entire model used in this research was to compare the results from this model to the previous model developed by Pemberton.

The simulation routine was verified by running traces for selected iterations and the results were compared to manual calculations. The simulation models the real world by actually producing impact points along a cut from the underlying normal error distributions. Then these impact points were ordered and the spaces between adjacent points and the runway is calculated and compared to the minimum takeoff width to determine a runway cut. These impact point spaces were calculated manually and compared to the minimum takeoff width. The determination of a runway cut agreed with the manual calculations. Next, the simulation routine was compared to the discrete approximation routine and the values from the discrete approximation were within the 99% confidence interval from the simulation results.

See columns P_{CL} and SIM in table II.

Next, the search routine was verified for obvious cases. When the minimum launch widths have large overlap, one intuitively expects that an aim point at the center of the runway is the "best" and this agrees with the model where, for a minimum launch window of 100 feet within a runway width of 150 feet the "best" strategy had one aim point in the center of the runway. Also, previous results by Pemberton were available to compare against this new program. See table II for comparison between Pemberton's model and RAM (Runway Attack Model) developed in this research. Although agreement is not 100%, it is felt that improvements in the calculation of the shortest runway length for the MLW, the increased efficiency of the discrete probability calculation and the new search algorithm are responsible for the minor differences in the number of weapons necessary to achieve the predetermined level of runway closure (0.8) and the estimated probability of runway closure.

The validation process discussed above is for the mode 1 operation in the program where the program searches for the "best" strategy to obtain the predetermined P_{CL} value. Modes 2 and 3, search for highest PC for fixed number of weapons and evaluation of any strategy, are subsets of mode 1 and are felt to be validated in the previous analysis.

Table II. Validation Results

	INPUTS									WEAPONS PER			
	RUNLEN	RUNWID	TOLEN	TOWID	W	CEP	P_{CL}	CUT LOCATIONS	AIM POINTS	AIM POINT	P_{CUT}	P_{CL}	SIM 99% CI
P	8000	150	2000	50	1000	100	.8	(1)	75	8	.96	.84	
R								(3)	75	7	.95	.82	(.74,.82)
P	8000	150	2000	50	250	20	.8	(2)	42,107	2,2	.99	.97	
R								(3)	36,75,114	1,1,1	.98	.91	(.84,.92)
P	8000	150	2000	100	250	100	.8	(1)	75	5	.96	.82	
R								(3)	75	5	.97	.89	(.84,.92)
P	8000	150	2000	50	1000	20	.8	(2)	35,75,114	1,1,1	1.0	.99	
R								(3)	44,106	1,1	.98	.94	(.88,.96)
P	8000	150	2000	100	250	20	.8	(2)	75	1	.99	.98	
R								(3)	75	1	.99	.98	(.94,1.0)
P	8000	150	2000	50	250	100	.8	(1)	35,75,115	3,5,3	.96	.81	
R								(3)	75	10	.96	.84	(.78,.86)
P	8000	150	2000	100	1000	20	.8	(2)	75	1	1.0	1.0	
R								(3)	75	1	1.0	1.0	(.96,1.0)

	INPUTS								WEAPONS			
	RUNLEN	RUNWID	TOLEN	TOWID	W	CEP	P_{CL}	CUT LOCATIONS	AIM POINTS	PER AIM POINT	P_{CUT}	P_{CL}	SIM 99% CI
P	8000	150	2000	100	1000	100	.8	(1)	75	4	.97	.85	
R								(3)	75	4	.97	.89	(.84,.92)

(1) - 1739, 3223, 4708, 6193, 7678

(2) - 1943, 3835, 5727, 7620

(3) - 1750, 3250, 4750, 6250

P - Results are from the Pemberton model

R - Results are from the RAM model

VII. Sensitivity Analysis

The task of sensitivity analysis is very much an art rather than a science. The sensitivity of this model to a change in one or more input variables depends on the actual values of these variables and on how big a change is proposed on selected variables. Since these factors are different for each problem, no automatic sensitivity routine was included in the model. A sensitivity analysis can be performed by changing the input variables and re-doing the problem. An example of the sensitivity of this model for one specific problem was performed by changing the input value of one factor at a time. The total number of weapons was chosen as the response variable. The first step was to run the program with the original input variables.

Input:			Output:
RUNLEN	-	8000	N = 12
RUNWID	-	150	
TOLEN	-	2000	
TOWID	-	50	
W	-	250	
CEP	-	20	
P_{CL}	-	0.8	

The second step was to vary one input variable at a time to find the range over which the response variable (total number of weapons) did not change, i.e., the response variable is insensitive to changes within this range for this input variable. No two way interactions, i.e., varying two input variables at one time, were analyzed because of the complexity of the task and the time available for this report. The results for the one factor sensitivity analysis for the example problem are:

Input Variable	Original Input Values	Insensitive Range
RUNLEN	8000	(7838, 9775)
RUNWID	150	(130, 155)
TOLEN	2000	(1646, 2040)
TOWID	50	(48, 57)
W	250	(200, 500)
CEP	20	(16, 23)

Therefore, a change in the length of the runway from 8000 to 7850 feet will not change the total number of weapons necessary to achieve at least a 0.8 probability of runway closure. But, a change in the minimum takeoff width from 50 to 45 feet will change the total number of weapons necessary to achieve at least a 0.8 probability of runway closure.

VIII. Conclusions and Recommendations

This research effort resulted in a computer program that can aid a commander or planner with the decision of how to "best" attack a runway. This program is an extension of the previous work by Pemberton (Ref 11). Improvements have been made in the efficiency of the computer program and an overall program was developed to offer the user three modes of operation.

Conclusions

This is the only runway attack program that does not require the user to have an initial attack strategy, but rather this program returns an attack strategy that is considered "best" to close the runway at a given level. This program allows a planner to see what one "best" strategy looks like. If the "best" strategy found in this program is not feasible due to limited number of weapons or tactical considerations, then using modes 2 and 3 iteratively will lead to a "best" attack strategy for a particular problem and estimate its probability of runway closure. Mode 2 allows the user to specify a fixed number of weapons available and returns the highest level of runway closure using the fixed number of weapons in a "best" way. Mode 3 allows the user to change the individual cut strategies, while leaving the number and locations of cuts the same as that identified from mode 1. to design a tactically feasible attack strategy. The program will evaluate this strategy and return the estimated level of runway closure for this independent strategy.

By using this program in a series of defining and re-defining the runway attack problem, the user will gain insights and experience about the problem of attacking runways.

Recommendations

While this research effort answered many of the questions posed in the beginning of this report, it raised and left unanswered other questions. Further work and research should be directed in these areas:

1. Adjust the computer program to address two other problem scenarios:

(a) R&D scenario, where the weapon yield and accuracy are the main factors in determining the "best" strategy.

(b) Runway planning scenario, where the vulnerability of a current or proposed base to enemy attack can be evaluated with the runway dimensions as the main factors determining the "best" strategy. The objective here is to minimize the highest probability of runway closure, subject to cost constraints.

2. Adapt this program to an interactive mode. The program is small enough to warrant use in the interactive, rather than batch mode.

3. Pursue the order statistics approach for the probability of cut presented in Chapter III.

4. Expand the discrete approximation routine to

include cases where s_1 and s_{NSPW-2} do not overlap.

5. Consider non-identical cuts in obtaining the "best" strategy to obtain a certain level of runway closure.

6. Add an automatic sensitivity analysis routine.

(a) Sensitivity can be in terms of changes to total number of weapons as in Chapter VII.

(b) Sensitivity analysis taking two factor and higher interactions into consideration.

(c) Analyzing the sensitivity of the "best" strategy to changes in input variables.

7. Consider non-circular error distributions for weapons in terms of REP and DEP.

Bibliography

1. David, H.A. Order Statistics. New York: John Wiley and Sons, Inc., 1970.

2. "The Durandal, a Runway Wrecker," Air Force Times, 43 (5):25 (August 23, 1982).

3. Eckler, A.R., "A Survey of Coverage Problems Associated with Point and Area Targets," Technometrics, 11:561-589 (1969).

4. Emerson, D.E. "AIDA: An Airbase Damage Assessment Model," RAND, R-1872-PR: (September 1976)

5. Emerson, D.E. "TSARINA: User's Guide to a Computer Model for Damage Assessment of Complex Air Base Targets," RAND, N-1460-AF: (July 1980).

6. Harter, Leon, Consultation with, Guest Lecturer, Math department, School of Engineering, Air Forct Institute of Technology, Wright-Patterson AFB, Ohio, 1982.

7. Jaiswal, N.K. and SANGAL, P.P. "Expected Damaged Area for Stick and Triangular Pattern Bombing," Operations Research, 20: 344-349 (1972).

8. Khazanie, Ramakant, Basic Probability Theory and Applications. Pacific Palisades, CA: Goodyear Publishing Company Inc., 1976.

9. Manz, Dr. Bruno J. "A Closed Form Solution for the Probability of Runway Closure by Cluster Munitions," Directorate of Aerospace Studies DCS/Plans and Programs, HQ AFSC, Kirtland AFB, New Mexico, June 1981.

10. Neu, Carl Richard, "Attacking Hardened Air Bases (AHAB): A Decision Analysis Aid for the Tactical Commander," RAND, R-1422-PR: (August 1974).

11. Pemberton, John C. "A Generalized Computer Model for the Targeting of Conventional Weapons to Destroy a Runway," MS thesis, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1980.

12. Seyb, E.K. and BLAAUW, F.H.M. "A Computer Program for the Calculation of Runway Interdiction Probabilities." Technical memorandum STC TM-431. SHAPE Technical Centre, The Hague, June 1974.

13. Shannon, Robert E. Systems Simulation: The Art and Science. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1975.

APPENDIX A

<u>User's Guide</u>

User's Guide to Runway Attack Model (RAM)

This is an expected value model that gives the "best" strategy to attack a runway. This program is written in Fortran 77, requires approximately 63,300 bytes of central memory (CDC 6600) and takes approximately 10 seconds of execution time to run an average problem. The execution time requirement increases to approximately 200 seconds when the simulation approximation subroutine is being used. The program uses calls to the International Mathematical and Statistics Libraries (IMSL) for procedures:

MDNOR - returns the cumulative normal distribution

GGNML - returns a normal random variate

VSRTA - returns a sorted array.

This program has three modes.

Mode 1

In this mode the user specifies the desired level of runway closure and the program returns the "best" strategy to achieve at least that level of runway closure.

Mode 2

In this mode the user specifies the number of weapons available and the program returns the "best" strategy to use and gives the expected level of runway closure.

Mode 3

In this mode the user specifies an independent strategy, using the number and location of cuts identified from mode 1, and the program returns the expected level of runway closure.

Input variables are:

Mode

1 - User specifies desired probability

2 - User specifies number of weapons available

3 - User specifies attack strategy

RUNLEN	- Runway length in feet
RUNWID	- Runway width in feet
TOLEN	- Minimum takeoff length in feet
TOWID	- Minimum takeoff width in feet
W	- Weapon yield in lbs. TNT
CEP	- Weapon accuracy in feet
P_{CL}	- Mode 1, desired probability of runway closure
NUMBER	- Mode 2, number of weapons available.

<u>Card Deck Setup for the CDC 6600</u>

```
job card
FTN5.
ATTACH,IMSL,ID=LIBRARY,SN=ASD.
LIBRARY,IMSL.
LGO.
*EOR
                Program
*EOR
                Data
*EOR
```

Data Cards:

Card 1   mode (1, 2, 3)

Card 2   RUNLEN, RUNWID, TOLEN, TOWID, W, CEP

Card 3   PCL or NUMBER

Card 3 and on (mode 3 operation specifying an independent strategy)

- number of cuts (as identified from mode 1)
- location of cut 1, number of aim points for cut 1
- aim point location 1, number of weapons for aim point 1
- aim point location 2, number of weapons for aim point 2

•
•
•

- location of cut 2, number of aim points for cut 2
- aim point location 1, number of weapons for aim point 1
- aim point location 2, number of weapons for aim point 2

•
•
•

- location of last cut, number of aim points for last cut
- aim point location 1, number of weapons for aim point location 1

- aim point location 2, number of weapons for
  aim point location 2.

Mode 1 example:

```
1
8000. 150. 2000. 50. 1000. 50.
0.8
*EOR
```

This card set up will return the "best" strategy for closing a runway (8,000 by 150 feet) for a MLW (2,000 by 50 feet) with weapons (1,000 lbs TNT, 50 foot CEP) with a desired level of closure 0.8. See figure A-1 for results.

Mode 2 example:

```
2
8000. 150. 2000. 50. 1000. 50.
15
*EOR
```

This card set up will return the "best" strategy for employing 15 weapons against the same runway as above. See figure A-2 for results.

Mode 3 example:

```
3
8000. 150. 2000. 50. 1000. 50.
4
1750.   1
    75.    4
3250.   2
    50.    3
```

```
        75.    2
    4750.   3
        25.    1
        75.    2
       100.    3
    6250.   4
        25.    1
        50.    2
        75.    2
       100.    1
    *EOR
```

This card set up will return the expected probability of runway closure for this independent strategy. Note, the number of cuts used and cut locations are identical to the output from mode 1 as shown in figure A-1. Note also, there is no requirement for symmetry for the aim point locations, or number of weapons per aim point for this mode of operation. See figure A-3 for results.

```
   RUNWAY       MINIMUM LAUNCH    WEAPON     PROBABILITY
                    WINDOW                        OF
LENGTH WIDTH  LENGTH WIDTH     YIELD     CEP   CLOSURE

 8000.  150.   2000.    50.      1000.    50.     .80


RUNWAY                   8000. BY     150. FEET

MIN LAUNCH WINDOW        2000. BY      50. FEET

WEAPON CHARACTERISTICS

     YIELD  1000.00 POUNDS

     CEP        50.   FEET

PROBABILITY OF CLOSURE   .83

TOTAL NUMBER OF WEAPONS   16

AIM POINTS        NUMBER OF

(LENGTH, WIDTH)  WEAPONS

1750.00   54.00     2
1750.00    6.00     2
3250.00   54.00     2
3250.00    6.00     2
4750.00   54.00     2
4750.00    6.00     2
6250.00   54.00     2
6250.00    6.00     2
```

Figure A-1. Mode 1 output

```
   RUNWAY       MINIMUM LAUNCH    WEAPON        NUMBER
                    WINDOW                        OF
LENGTH WIDTH  LENGTH WIDTH     YIELD     CEP  WEAPONS

 8000.  150.   2000.   50.     1000.     50.     15

RUNWAY                   8000. BY     150. FEET

MIN LAUNCH WINDOW      2000. BY      50. FEET

WEAPON CHARACTERISTIC

     YIELD  1000.00 POUNDS

     CEP       50.   FEET

PROBABILITY OF CLOSURE   .76

TOTAL NUMBER OF WEAPON    15

AIM POINTS        NUMBER OF

(LENGTH, WIDTH)  WEAPONS

1750.00    54.00      2
1750.00     6.00      2
3250.00    54.00      2
3250.00    96.00      2
4750.00    54.00      2
4750.00     6.00      2
6250.00    56.00      1
6250.00    75.00      1
6250.00    94.00      1
```

Figure A-2. Mode 2 output

```
   RUNWAY      MINIMUM LAUNCH   WEAPON
                   WINDOW
LENGTH WIDTH  LENGTH WIDTH     YIELD     CEP

 8000.  150.   2000.   50.     1000.     50.


INDEPENDENT ANALYSIS

INDEPENDENT STRATEGY:

CUT         AIM POINT   NUMBER PER
LOCATION    LOCATION    AIM POINT

   1750.
                75.00         4

   3250.
                50.00         3
                75.00         2

   4750.
                25.00         1
                75.00         2
               100.00         3

   6250.
                25.00         1
                50.00         1
                75.00         2
               100.00         1

PROBABILITY OF RUNWAY CLOSURE IS  .90
```

Figure A-3. Mode 3 output

APPENDIX B

Relationship Between CEP and Sigma

Relationship Between CEP and Sigma

The weapon error distribution in this research is assumed to be circular normal, centered at the aim point. The circular normal probability density function is:

$$f(x,y) = \frac{1}{2\pi\sigma^2} \exp\left\{-\tfrac{1}{2}\,\frac{x^2+y^2}{\sigma^2}\right\} \qquad \text{(B-1)}$$

(Ref 11: 60)

The probability that the weapon will land in a circle with radius R and area A is given by:

$$P(A) = \frac{1}{2\pi\sigma^2} \iint_A \exp\left\{-\tfrac{1}{2}\left(\frac{x^2+y^2}{2}\right)\right\} dx\, dy \qquad \text{(B-2)}$$

Transfering to polar coordinates by letting

$$x = r \cos \theta$$

$$y = r \sin \theta$$

yields

$$P(A) = \frac{1}{2\pi\sigma^2} \int_0^R \int_0^2 \exp\left\{-\tfrac{1}{2}\left(\frac{r^2}{\sigma^2}\right)\right\} r\, dr\, d\theta \qquad \text{(B-3)}$$

$$= \int_0^R \frac{1}{\sigma^2}\left[\frac{1}{2\pi}\int_0^2 d\theta\right] r \exp\left\{-r^2/2\sigma^2\right\} dr$$

$$= \int_0^R r/\sigma^2 \exp\left\{-r^2/2\sigma^2\right\} dr$$

$$= 1 - \exp\left\{-R^2/2\sigma^2\right\}$$

CEP is defined as the radius of a circle with a 0.5 probability of a weapon landing within it, or:

$$P(A) = 0.5 = 1 - \exp -\left\{CEP^2/2\sigma^2\right\} \qquad \text{(B-4)}$$

moving terms from one side to the other yields

$$1 - 0.5 = \exp\left\{-\mathrm{CEP}^2/\ 2\sigma^2\right\}$$

$$0.5 = \exp\left\{-\mathrm{CEP}^2/2\sigma^2\right\}$$

$$\ln(0.5) = -\mathrm{CEP}^2/2\sigma^2$$

$$-\ln(0.5) = \mathrm{CEP}^2/2\sigma^2$$

$$\ln(2) = \tfrac{1}{2}(\mathrm{CEP}^2/2\sigma^2)$$

$$2\ln(2) = (\mathrm{CEP}/\sigma)^2$$

$$\sqrt{2\ln(2)} = \mathrm{CEP}/\sigma$$

$$\sqrt{2\ln(2)}\ \sigma = \mathrm{CEP}$$

$$\sigma = \mathrm{CEP}/\sqrt{2\ln(2)}$$

or

$$\sigma = \mathrm{CEP}/1.1774 \qquad \text{(B-5)}$$

This relationship allows converting the normal units of weapon accuracy, CEP, into sigma, which is used in the research.

APPENDIX C

Development of Shortest Runway Length

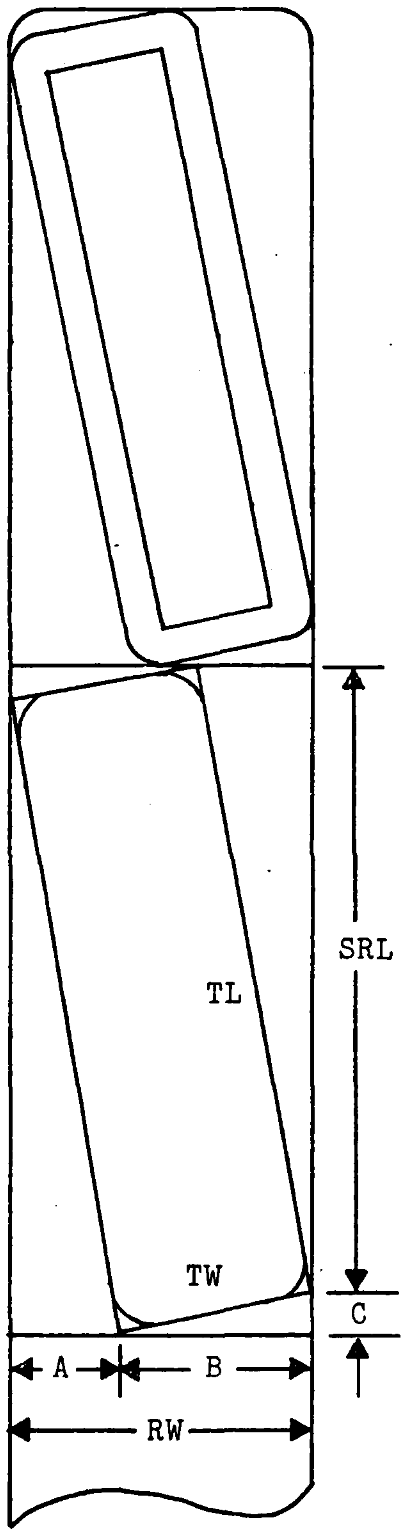
Development of Shortest Runway Length

The determination of the minimum number of cuts required to close a runway involves finding the shortest runway length that completely contains a minimum launch window (MLW, takeoff length by takeoff width).

When the weapon damage radius is taken into consideration, the dimensions for the runway and the MLW are increased by an amount equal to the damage radius all around. This presents a problem because not the shapes are not rectangular, rather, they have corners that are described by an arc with radius equal to the damage radius.

In order to find the shortest runway length that sompletely contains a MLW, the MLW was approximated by a rectangle of dimensions effective takeoff width by effective takeoff length. The shortest distance was approximated by the base of the triangle formed by the effective takeoff length and the line prependicular to the runway that touches the corner of the MLW. This distance was chosen because it under-estimates the actual distance. The distance from one prependicular to the next prependicular overestimates the actual distance. The underestimation is preffered because it is better to have too many cuts rather than too few.

This estimate, called the shortest runway length (SRL) can be found by using a geometric argument. From figure B-1, RW can be divided into two segments A and B.



RW - Effective Runway Width
TL - Effective Takeoff Length
TW - Effective Takeoff Width
SRL - Shortest Runway Length

Figure C-1. Shortest Runway Length

The triangle A, SRL, TL is similar to the triangle C, TW, B. Since these two triangles are similar, the ratios

$$TL/SRL = TW/B \tag{B-1}$$

are equal. Another relationship is

$$(TL)^2 = (SRL)^2 + (RW-B)^2 \tag{B-2}$$

We now have two equations in two unknowns, SRL and B. Solving for B from equation (B-1) yields

$$B = SRL(TW/TL) \tag{B-3}$$

Substituting equation (B-3) into equation (B-2) yields

$$(TL)^2 = (SRL)^2 + (RW-SRL(TW/TL))^2 \tag{B-4}$$

squaring the terms yields

$$(TL)^2 = (SRL)^2 + (RW)^2 - 2(SRL)(RW)(TW)/(TL) + (SRL)^2(TW)^2/(TL)^2 \tag{B-5}$$

collecting terms yields

$$\left[1+(TW)^2/(TL)^2\right](SRL)^2 - \left[2(RW)(TW)/(TL)\right] SRL + \left[(RW)^2 - (TL)^2\right] = 0 \tag{B-6}$$

applying the quadratic formula yields

$$\frac{\left[2(RW)(TW)/TL\right] \pm \sqrt{\left[2(RW)(TW)/TL\right]^2 - (4)\left[1+\frac{TW^2}{TL^2}\right]\left[RW^2 - TL^2\right]}}{2(1+TW^2/TL^2)} \tag{B-7}$$

The determinant can be simplified as

$$4(RW)^2(TW)^2/TL^2 - 4\left[RW^2+RW^2TW^2/TL^2 - TL^2 - TW^2\right]$$
$$= -4(RW)^2 + 4(TL)^2 + 4(TW)^2$$
$$= -4\left[RW^2 - TL^2 - TW^2\right]$$
$$= 4\left[TL^2 + TW^2 - RW^2\right] \qquad \text{(B-8)}$$

Substituting (B-8) into (B-7) we have

$$SRL = \frac{2(RW)(TW)/TL \pm 2\sqrt{TL^2 + TW^2 - RW^2}}{2\left[1 + TW^2/TL^2\right]} \qquad \text{(B-9)}$$

cancelling the 2 and expanding the denominator yields

$$SRL = \frac{(RW)(TW)/TL \pm\sqrt{TL^2 + TW^2 - RW^2}}{\left[\frac{TL^2 + TW^2}{TL^2}\right]} \qquad \text{(B-10)}$$

Multiplying top and bottom by TL^2 yields

$$SRL = \frac{TL^2\left[(RW)(TW)/TL\right] \pm TL^2\sqrt{TL^2 + TW^2 - RW^2}}{TL^2 + TW^2} \qquad \text{(B-11)}$$

cancelling one TL in the first term yields

$$SRL = \frac{(RW)(TL)(TW) \pm TL^2\sqrt{TL^2 + TW^2 - RW^2}}{TL^2 + TW^2} \qquad \text{(B-12)}$$

For example, if the effective takeoff width was equal to the effective runway width, then the shortest runway length would just be the effective takeoff length.

In this example, TW = RW

and

$$SRL = \frac{(RW)(TL)(RW) \pm TL^2\sqrt{TL^2 + RW^2 - RW^2}}{TL^2 + RW^2}$$

$$= \frac{(RW)^2(TL) \pm TL^2\sqrt{TL^2}}{TL^2 + RW^2}$$

$$= \frac{(RW)^2(TL) \pm TL^3}{TL^2 + RW^2}$$

taking the positive root,

$$= \frac{(RW)^2(TL) + TL^3}{TL^2 + RW^2}$$

$$= \frac{TL\left[TL^2 + RW^2\right]}{TL^2 + RW^2}$$

$$= TL$$

The solution is real as long as the determinant

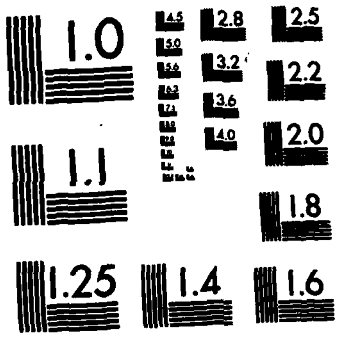
$$TL^2 + TW^2 - RW^2$$

is positive. Geometrically, this is saying that this procedure works for cases where the diagonal of the minimum launch window is greater than the effective runway width. If this were not so, the MLW could be positioned vertically on the runway.

APPENDIX D

Computer Listing

```
      PROGRAM RAM
      DIMENSION NUMBOM(20),AIMLOC(20)
      DIMENSION NPERCT(20)
      COMMON PROSEC(20,20,2)
      COMMON /INPUT/ RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
      COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
      COMMON /GRP/ NSPW,STEP
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
      COMMON /BEST/ PCBEST(20),NALBST(20),NBMBST(20,5),AIMBST(20,5)
      COMMON /SIMS/ DSEED,NITERA
      COMMON SIMCAL
      DOUBLE PRECISION DSEED
      LOGICAL SIMCAL
C
      READ*,MODE
      READ*,RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
C
      CALL INIT
C
      CALL CONVRT
C
      IF (MODE.EQ.1) THEN
         READ*,PCLOSE
         PRINT 100
         PRINT 110
         PRINT 120
         PRINT 130,RUNLEN,RUNWID,TOLEN,TOWID,W,CEP,PCLOSE
         CALL RUNWAY (PCLOSE)
         CALL BOUNDS (MIN,MAX)
         CALL SEARCH (MIN,MAX,NUMB,PC)
         DO 10 I=1,NCUTS
               NPERCT(I)=NUMB
10       CONTINUE
         CALL EVAL (NPERCT,NUMBER,PCLEST)
         CALL RESLTS (NPERCT,NUMBER,PCLEST)
         STOP
C
      ELSE IF (MODE.EQ.2) THEN
         READ*,NUMBER
         PRINT 102
         PRINT 110
         PRINT 122
         PRINT 132,RUNLEN,RUNWID,TOLEN,TOWID,W,CEP,NUMBER
         CALL RUNWAY (1.0)
         CALL NUMSET (NPERCT,NUMBER,PCLEST)
         CALL RESLTS (NPERCT,NUMBER,PCLEST)
         STOP
      ELSE IF (MODE.EQ.3) THEN
         PRINT 103
         PRINT 113
         PRINT 123
         PRINT 130,RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
         CALL RUNWAY (1.0)
         CALL INDEP
```


1.0
4.5
5.0
5.6
6.3
7.1
8.0
9.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

```
         STOP
C
      END IF
C
      PRINT 140
C
C
C
100   FORMAT('1',3X,'RUNWAY',5X,'MINIMUM LAUNCH',
     Z3X,'WEAPON',4X,'PROBABILITY')
102   FORMAT('1',3X,'RUNWAY',5X,'MINIMUM LAUNCH',
     Z3X,'WEAPON',6X,'NUMBER')
103   FORMAT('1',3X,'RUNWAY',5X,'MINIMUM LAUNCH',
     Z3X,'WEAPON')
110   FORMAT(' ',T19,'WINDOW',T47,'OF')
113   FORMAT(' ',T19,'WINDOW')
120   FORMAT(' LENGTH WIDTH  LENGTH WIDTH',4X,
     Z'YIELD    CEP',2X,'CLOSURE')
122   FORMAT(' LENGTH WIDTH  LENGTH WIDTH',4X,
     Z'YIELD    CEP',2X,'WEAPONS')
123   FORMAT(' LENGTH WIDTH  LENGTH WIDTH',4X,
     Z'YIELD    CEP')
130   FORMAT('0',F6.0,F6.0,2X,F6.0,F6.0,4X,F5.0,3X,F4.0,3X,F4.2)
132   FORMAT('0',F6.0,F6.0,2X,F6.0,F6.0,4X,F5.0,3X,F4.0,3X,I4)
140   FORMAT('0***   INPUT ERROR - CHECK MODE SPECIFIED   ***')
      END
      SUBROUTINE INIT
      COMMON /SIMS/ DSEED,NITERA
      DOUBLE PRECISION DSEED
C
      DSEED=585285124.
      NITERA=2770
      RETURN
      END
      SUBROUTINE CONVRT
      COMMON /INPUT/ RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
      COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
      COMMON /GRP/ NSPW,STEP
      COMMON SIMCAL
      LOGICAL SIMCAL
C
      DAMRAD=3.54*((W)**(1./3.))
      STADEV=(CEP)/1.1774
C
      STEP=5.0
      ERUNLE=RUNLEN+2.*DAMRAD
      ERUNWI=RUNWID+2.*DAMRAD
      ETOLEN=TOLEN +2.*DAMRAD
      ETOWID=TOWID +2.*DAMRAD
      SIMCAL=.FALSE.
      IF((ERUNWI-2*ETOWID).GT.(2*STEP)) THEN
         SIMCAL=.TRUE.
```

```
            PRINT 100
            PRINT 110
            PRINT 120
         END IF
         NSPW=((ERUNWI-ETOWID)/STEP)+.0001
100      FORMAT('0*** SIMULATION APPROXMATION IN USE ***')
110      FORMAT('0*** RUN TIME MAY NEED TO BE INCREASED ***')
120      FORMAT('0*** RESULTS ARE ONLY ACCURATE TO +/- .01 ***')
         RETURN
         END
         SUBROUTINE PRONE(NUMAIM,NUMBOM,AIMLOC,PC)
         DIMENSION NUMBOM(20),AIMLOC(20),DAMLOC(20)
         COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
         COMMON /GRP/ NSPW,STEP
C
         DAMLOC(1)=AIMLOC(1)+DAMRAD
         BR=(ETOWID-DAMLOC(1))/STADEV
         BL=(ERUNWI-ETOWID-DAMLOC(1))/STADEV
         IF(BL.GT.BR) THEN
               PRINT*,'S1 AND SN DO NOT OVERLAP...DOUBLE CHECK INPUTS'
               PC=0.0
               RETURN
         END IF
         CALL MDNOR(BR,BONDP)
         CALL MDNOR(BL,BONDL)
         PC=BONDR-BONDL
         RETURN
         END
         SUBROUTINE PROCAL(NUMAIM,AIMLOC)
         DIMENSION AIMLOC(20),DAMLOC(20)
         COMMON PROSEC(20,20,2)
         COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
         COMMON /GRP/ NSPW,STEP
C
         DO 10 J=1,NUMAIM
               DAMLOC(J)=AIMLOC(J)+DAMRAD
10       CONTINUE
         DO 50 I1=1,NUMAIM
               DO 40 I2=1,NSPW
                     X=I2-1
                     BL=(X*STEP-DAMLOC(I1))/STADEV
                     BR=(X*STEP+ETOWID-DAMLOC(I1))/STADEV
                     CALL MDNOR(BL,PRL)
                     CALL MDNOR(BR,PRR)
                     PRR=1.-PRR
                     PROSEC(I1,I2,1)=PRL
                     PROSEC(I1,I2,2)=PRR
40             CONTINUE
50       CONTINUE
         RETURN
         END
```

```
      SUBROUTINE UNION(NUMAIM,NUMBOM,AIMLOC,PC)
      DIMENSION NUMBOM(20),AIMLOC(20)
      DIMENSION S(2,20), S1(2), S2(6)
      COMMON PROSEC(20,20,2)
      COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
      COMMON /GRP/ NSPW,STEP
C
      CALL PROCAL(NUMAIM,AIMLOC)
      TOTAL=0.0
      NM1=NSPW-1
      GAP=(ERUNWI-2*ETOWID)
      IF(GAP.LE.0.0) GO TO 50
      IF(GAP.LE.STEP) GO TO 30
      IF(GAP.LE.(2*STEP)) GO TO 10
      PRINT*,'******** USE SIMULATION ********'
      RETURN
10    CONTINUE
      DO 15 I=1,6
            S2(I)=1.0
15    CONTINUE
      DO 20 I1=1,NUMAIM
            SL1=PROSEC(I1,1,1)
            SL2=PROSEC(I1,2,1)
            SRNM1=PROSEC(I1,NM1,2)
            SRN=PROSEC(I1,NSPW,2)
            SM1NM1=PROSEC(I1,NM1,1) - (1.-PROSEC(I1,1,2))
            SM2N=PROSEC(I1,NSPW,1) - (1.-PROSEC(I1,2,2))
            SM2NM1=PROSEC(I1,NM1,1) - (1.-PROSEC(I1,2,2))
            IF (SM2NM1.LE. 0.) SM2NM1=0.
            SM3N=PROSEC(I1,NSPW,1) - (1.-PROSEC(I1,3,2))
            IF(SM3N .LE. 0.) SM3N=0.
            S2(1) = ((SL1 + SM1NM1 + SRNM1)**NUMBOM(I1))*S2(1)
            S2(2) = ((SL2 + SM2N + SRN)    **NUMBOM(I1))*S2(2)
            S2(3) = ((SL1 + SM2NM1 + SRNM1)**NUMBOM(I1))*S2(3)
            S2(4) = ((SL2 + SM3N + SRN)    **NUMBOM(I1))*S2(4)
            S2(5) = ((SL1 + SM1NM1 + SRN)  **NUMBOM(I1))*S2(5)
            S2(6) = ((SL1 + SM2NM1 + SRN)  **NUMBOM(I1))*S2(6)
20    CONTINUE
      TOTAL=TOTAL-S2(1)-S2(2)+S2(3)+S2(4)+S2(5)-S2(6)
30    CONTINUE
      S1(1)=1.0
      S1(2)=1.0
      DO 40 I1=1,NUMAIM
            SL=PROSEC(I1,1,1)
            SM1=PROSEC(I1,NSPW,1) - (1.-PROSEC(I1,1,2))
            SM2=PROSEC(I1,NSPW,1) - (1.-PROSEC(I1,2,2))
            SR=PROSEC(I1,NSPW,2)
            IF(SM2 .LE. 0.) SM2=0.
            S1(1) = ((SL + SM1 + SR) ** NUMBOM(I1)) * S1(1)
            S1(2) = ((SL + SM2 + SR) ** NUMBOM(I1)) * S1(2)
40    CONTINUE
            TOTAL=TOTAL-S1(1)+S1(2)
50    CONTINUE
```

```
      DO 55 I=1,NSPW
            S(1,I)=1.0
            S(2,I)=1.0
55    CONTINUE
      DO 70 I2=1,NSPW
            DO 60 I1=1,NUMAIM
      S(1,I2)=((PROSEC(I1,I2,1)+PROSEC(I1,I2,2))**NUMBOM(I1))*
     1S(1,I2)
            IF(I2.EQ.NSPW) GO TO 60
            I3=I2+1
      S(2,I2)=((PROSEC(I1,I2,1)+PROSEC(I1,I3,2))**NUMBOM(I1))*
     1S(2,I2)
60          CONTINUE
70    CONTINUE
      DO 80 I2=1,NSPW
            TOTAL=TOTAL+S(1,I2)
            IF(I2.EQ.NSPW) GO TO 80
            I3=I2+1
            TOTAL=TOTAL-S(2,I2)
80    CONTINUE
      PC=1.-TOTAL
      RETURN
      END
      SUBROUTINE SIM(NUMAIM,NUMBOM,AIMLOC,PC)
      DIMENSION NUMBOM(20),AIMLOC(20),DAMLOC(20)
      DIMENSION HIT(50),R1(1)
      COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
      COMMON /SIMS/ DSEED,NITERA
      DOUBLE PRECISION DSEED
C
      OPEN=0.0
      INT=1
      DO 10 I=1,NUMAIM
            DAMLOC(I)=AIMLOC(I)+DAMRAD
10    CONTINUE
      DO 200 ITER=1,NITERA
            NUMB=0
            DO 20 J1=1,NUMAIM
                  N1=NUMBOM(J1)
                  DO 20 J2=1,N1
                        NUMB=NUMB+1
                        CALL GGNML(DSEED,INT,R1)
                        HIT(NUMB)=DAMLOC(J1)+(STADEV*R1(INT))
20          CONTINUE
            CALL VSRTA(HIT,NUMB)
            NSHORT=0
            NLONG=0
            NSHLO=0
            DO 30 L=1,NUMB
                  IF(HIT(L).LT.0.) NSHORT=NSHORT+1
                  IF(HIT(L).GT.ERUNWI) NLONG=NLONG+1
30          CONTINUE
            IF(NSHORT.EQ.NUMB) GO TO 100
            IF(NLONG.EQ.NUMB) GO TO 100
```

```
          NSHLO=NSHORT+NLONG
          IF(NSHLO.EQ.NUMB) GO TO 100
C
C         IS THE FIRST WEAPON CLOSE TO THE RUNWAY EDGE?
C
          NFIRST=NSHORT+1
          IF(HIT(NFIRST).GT.ETOWID) GO TO 100
C
C         IS THE LAST WEAPON CLOSE TO THE RUNWAY EDGE?
C
          NLAST=NUMB-NLONG
          IF(HIT(NLAST).LT.(ERUNWI-ETOWID)) GO TO 100
C
C         IS THIS THE ONLY WEAPON ON THE RUNWAY
C
          IF(NFIRST.EQ.NLAST) GO TO 200
C
C         CHECK ON THE DISTANCE BETWEEN ADJACENT
C         IMPACT POINTS ON THE RUNWAY.
C
          NLM1=NLAST-1
          DO 40 J=NFIRST,NLM1
               DIS=HIT(J+1)-HIT(J)
               IF(DIS.GT.ETOWID) GO TO 100
40        CONTINUE
C
C         NO OPEN SPACE FOUND
C
          GO TO 200
100   OPEN=OPEN+1.0
200   CONTINUE
      XITER=NITERA
      TOTAL=OPEN/XITER
      PC=1.-TOTAL
      RETURN
      END
      SUBROUTINE RUNWAY(PCLOSE)
      COMMON /INPUT/ RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
      COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
C
      SRL=((ERUNWI*ETOLEN*ETOWID)+(ETOLEN**2.)*
     Z    SQRT(ETOLEN**2.+ETOWID**2.-ERUNWI**2.))/
     Z    (ETOLEN**2.+ETOWID**2.)
      CRUNLE=ERUNLE-(6.*STADEV)
      BETLEN=SRL-(6.*STADEV)
      CUTS=(CRUNLE/BETLEN)-1.
      NCUTS=INT(CUTS)
      REM=CUTS-NCUTS
      IF(REM .GT. 0.01) NCUTS=NCUTS+1
      PCSTAR=PCLOSE**(1./NCUTS)
      SPACE=(ERUNLE-ETOLEN)/(NCUTS)
      OVRLAP=(ETOLEN-SPACE)/2.
      CUTLOC(1)=OVRLAP+SPACE-DAMRAD
      DO 100 I=2,NCUTS
         CUTLOC(I)=CUTLOC(I-1)+SPACE
100   CONTINUE
      RETURN
      END
      SUBROUTINE BOUNDS(MIN,MAX)
      DIMENSION NUMBOM(20),AIMLOC(20)
      COMMON /INPUT/ RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
      COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
      COMMON /GRP/ NSPW,STEP
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
      COMMON SIMCAL
      LOGICAL SIMCAL
C
      MIN=ERUNWI/ETOWID
C
      NUMAIM=MIN
      SPACE=RUNWID/(MIN+1)
      DO 20 I1=1,20
         MAX=0
         DO 10 I2=1,NUMAIM
            NUMBOM(I2)=I1
            AIMLOC(I2)=I2*SPACE
            MAX=MAX+I1
10       CONTINUE
         IF(NUMAIM .EQ. 1 .AND. NUMBOM(1) .EQ. 1) THEN
            CALL PRONE(NUMAIM,NUMBOM,AIMLOC,PC)
            IF (PC .GE. PCSTAR) GO TO 90
         ELSE IF(SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIMLOC,PC)
            IF(PC.GE.PCSTAR+.01) GO TO 90
         ELSE IF(.NOT.SIMCAL) THEN
            CALL UNION(NUMAIM,NUMBOM,AIMLOC,PC)
            IF (PC .GE. PCSTAR) GO TO 90
         END IF
20    CONTINUE
90    RETURN
      END
      SUBROUTINE SEARCH (MIN,MAX,NUMB,PC)
      DIMENSION AIM(20),AIMO(20),AIMI(20),AIMLOC(20),NUMBOM(20)
      DIMENSION LAIM(5),IAIM(5)
      DIMENSION NBMINT(20),AIMINT(20)
      COMMON /INPUT/ RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
      COMMON /TRANS/ ERUNLE,ERUNWI,ETOLEN,ETOWID,DAMRAD,STADEV
      COMMON /GRP/ NSPW,STEP
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
      COMMON /BEST/ PCBEST(20),NALBST(20),NBMBST(20,5),AIMBST(20,5)
      COMMON SIMCAL
      LOGICAL SIMCAL
C
      CNTR=RUNWID/2.
      LCNTR=CNTR+1.
      TOL=0.0
      IF(SIMCAL) TOL=0.01
C
C
      DO 900 LOOP = MIN, MAX
      NUMB=LOOP
C
100   NUMAIM=1
      NUMBOM(1)=NUMB
      AIMLOC(1)=CNTR
      IF (NUMB .EQ. 1) THEN
         CALL PRONE(NUMAIM,NUMBOM,AIMLOC,PCBEST(1))
         NALBST(1)=NUMAIM
         NBMBST(1,1)=NUMBOM(1)
         AIMBST(1,1)=AIMLOC(1)
         GO TO 800
      END IF
      IF(SIMCAL) THEN
         CALL SIM(NUMAIM,NUMBOM,AIMLOC,PCBEST(NUMB))
      ELSE
         CALL UNION(NUMAIM,NUMBOM,AIMLOC,PCBEST(NUMB))
      END IF
      NALBST(NUMB)=NUMAIM
      NBMBST(NUMB,1)=NUMBOM(1)
      AIMBST(NUMB,1)=AIMLOC(1)
C
200   NUMAIM=2
      NREM=MOD(NUMB,2)
      IF(NREM .EQ. 1) GO TO 300
      PC2=0.0
      AIM(1)=0.0
      AIM(2)=RUNWID
      NUMBOM(1)=NUMB/2.
      NUMBOM(2)=NUMB/2.
C
      DO 230 J1=1,LCNTR,5
         AIMLOC(1)=J1-1.
         AIMLOC(2)=RUNWID-(J1-1.)
         IF(SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIMLOC,PCN)
         ELSE
            CALL UNION(NUMAIM,NUMBOM,AIMLOC,PCN)
         END IF
         IF(PCN .GT. PC2) GO TO 220
         LAST=J1-10
         PC2=0.0
         DO 210 J2=LAST,J1,1
            AIMLOC(1)=J2
            AIMLOC(2)=RUNWID-J2
            IF(SIMCAL) THEN
               CALL SIM(NUMAIM,NUMBOM,AIMLOC,PCN)
            ELSE
               CALL UNION(NUMAIM,NUMBOM,AIMLOC,PCN)
```

```
                END IF
                IF(PCN .LE. PC2) GO TO 240
                PC2=PCN
                AIM(1)=AIMLOC(1)
                AIM(2)=AIMLOC(2)
210          CONTINUE
220          PC2=PCN
             AIM(1)=AIMLOC(1)
             AIM(2)=AIMLOC(2)
230       CONTINUE
240       DIF=PC2-PCBEST(NUMB)
          IF(DIF.GT.TOL) THEN
             PCBEST(NUMB)=PC2
             NALBST(NUMB)=NUMAIM
             NBMBST(NUMB,1)=NUMBOM(1)
             NBMBST(NUMB,2)=NUMBOM(2)
             AIMBST(NUMB,1)=AIM(1)
             AIMBST(NUMB,2)=AIM(2)
          ELSE
             GO TO 800
          END IF
          IF(NUMB .EQ. 2) GO TO 800
C
300       NUMAIM=3
          LBOMB=(NUMB-1)/2
          PC3=0.0
          PCINT=0.0
          DO 360 NBOM=1,LBOMB
             NUMBOM(1)=NBOM
             NUMBOM(2)=NUMB-2*NBOM
             NUMBOM(3)=NBOM
          AIM(1)=0.0
          AIM(2)=CNTR
          AIM(3)=RUNWID
          DO 330 J1=1,LCNTR,5
             AIMLOC(1)=J1-1.
             AIMLOC(2)=CNTR
             AIMLOC(3)=RUNWID-(J1-1)
             IF(SIMCAL) THEN
                CALL SIM(NUMAIM,NUMBOM,AIMLOC,PCN)
             ELSE
                CALL UNION(NUMAIM,NUMBOM,AIMLOC,PCN)
             END IF
             DIF=PCN-PC3
             IF(DIF.GT.TOL) GO TO 320
             LAST=J1-10.
             PC3=0.0
             DO 310 J2=LAST,J1,1
                AIMLOC(1)=J2
                AIMLOC(2)=CNTR
                AIMLOC(3)=RUNWID-J2
                IF(SIMCAL) THEN
                   CALL SIM(NUMAIM,NUMBOM,AIMLOC,PCN)
                ELSE
```

```
                 CALL UNION(NUMAIM,NUMBOM,AIMLOC,PCN)
               . END IF
                    DIF=PCN-PC3
                    IF(DIF.LE.TOL) GO TO 340
                 PC3=PCN
                 AIM(1)=AIMLCC(1)
                 AIM(2)=AIMLOC(2)
                 AIM(3)=AIMLOC(3)
310           CONTINUE
320           PC3=PCN
              AIM(1)=AIMLOC(1)
              AIM(2)=AIMLOC(2)
              AIM(3)=AIMLOC(3)
330        CONTINUE
340        IF(PC3.GT.PCINT) THEN
              PCINT=PC3
              NALINT=NUMAIM
              DO 350 I=1,3
                 NBMINT(I)=NUMBOM(I)
                 AIMINT(I)=AIM(I)
350           CONTINUE
           END IF
360        CONTINUE
           IF(PCINT.GT.PCBEST(NUMB)) THEN
              PCBEST(NUMB)=PCINT
              NALBST(NUMB)=NALINT
              DO 370 I=1,3
                 NBMBST(NUMB,I)=NBMINT(I)
                 AIMBST(NUMB,I)=AIMINT(I)
370           CONTINUE
           ELSE
              GO TO 800
           END IF
           IF(NUMB .EQ. 3) GO TO 800
C
400        NUMAIM=4
           NREM=MOD(NUMB,4)
           IF(NREM .EQ. 1 .OR. NREM .EQ. 3) GO TO 500
           PC4=0.0
           PCINT=0.0
           LBOMB=(NUMB-2)/2
           DO 493 NBM=1,LBOMB
           SPACE=RUNWID/(NUMAIM+1)
           NUMBOM(1)=NBM
           NUMBOM(4)=NBM
           LBMB2=(NUMB-2*NBM)/2
           DO 493 NBM2=1,LBMB2
              NUMBOM(2)=NUMB-2*NBM
              NUMBOM(3)=NUMB-2*NBM
           DO 410 I=1,NUMAIM
              AIM(I)=I*SPACE
              LAIM(I)=AIM(I)
410        CONTINUE
           IF(SIMCAL) THEN
```

```
            CALL SIM(NUMAIM,NUMBOM,AIM,PC4)
         ELSE
            CALL UNION(NUMAIM,NUMBOM,AIM,PC4)
         END IF
 430     STEP1=5.
         AIMI(1)=AIM(1)
         AIMI(4)=AIM(4)
         AIMO(1)=AIM(1)
         AIMO(4)=AIM(4)
C
C MOVE INNER PAIR OF AIM POINTS IN OR OUT
C
 440     AIMI(2)=AIM(2)+STEP1
         AIMI(3)=AIM(3)-STEP1
         IF(SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIMI,PCI)
         ELSE
            CALL UNION(NUMAIM,NUMBOM,AIMI,PCI)
         END IF
         AIMO(2)=AIM(2)-STEP1
         AIMO(3)=AIM(3)+STEP1
         IF(SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIMO,PCO)
         ELSE
            CALL UNION(NUMAIM,NUMBOM,AIMO,PCO)
         END IF
         DIF1=PCO-PC4
         DIF2=PC4-PCI
         DIF3=PCI-PC4
         DIF4=PC4-PCO
         IF(DIF1.GT.TOL .AND. DIF2.GT.TOL) THEN
            PC4=PCO
            AIM(2)=AIMO(2)
            AIM(3)=AIMO(3)
         ELSE IF(DIF3.GT.TOL .AND. DIF4.GT.TOL) THEN
            PC4=PCI
            AIM(2)=AIMI(2)
            AIM(3)=AIMI(3)
         ELSE IF(DIF4.GT.TOL .AND. DIF2.GT.TOL) THEN
            IF(STEP1 .LE. 0.5) GO TO 450
            STEP1=STEP1/2.
            GO TO 440
         END IF
C
C MOVE OUTER PAIR OF AIM POINTS IN OR OUT
C
 450     AIMI(2)=AIM(2)
         AIMI(3)=AIM(3)
         AIMO(2)=AIM(2)
         AIMO(3)=AIM(3)
         STEP2=5.
 460     AIMI(1)=AIM(1)+STEP2
         AIMI(4)=AIM(4)-STEP2
         IF(SIMCAL) THEN
         CALL SIM(NUMAIM,NUMBOM,AIMI,PCI)
      ELSE
         CALL UNION(NUMAIM,NUMBOM,AIMI,PCI)
      END IF
      AIMO(1)=AIM(1)-STEP2
      AIMO(4)=AIM(4)+STEP2
      IF(SIMCAL) THEN
         CALL SIM(NUMAIM,NUMBOM,AIMO,PCO)
      ELSE
         CALL UNION(NUMAIM,NUMBOM,AIMO,PCO)
      END IF
      DIF1=PCO-PC4
      DIF2=PC4-PCI
      DIF3=PCI-PC4
      DIF4=PC4-PCO
      IF(DIF1.GT.TOL .AND. DIF2.GT.TOL) THEN
         PC4=PCO
         AIM(1)=AIMO(1)
         AIM(4)=AIMO(4)
      ELSE IF(DIF3.GT.TOL .AND. DIF4.GT.TOL) THEN
         PC4=PCI
         AIM(1)=AIMI(1)
         AIM(4)=AIMI(4)
      ELSE IF(DIF4.GT.TOL .AND. DIF2.GT.TOL) THEN
         IF(STEP2.LE.0.5) GO TO 470
         STEP2=STEP2/2.
         GO TO 460
      END IF
C
C ROUND AIM POINT LOCATIONS TO NEAREST FOOT.
C
470   CONTINUE
      DO 480 I=1,4
         I  M(I)=AIM(I)+0.5
480   CONTINUE
C
C CHECK MOVEMENT FROM LAST ADJUSTMENT.
C
      IF(LAIM(1).EQ.IAIM(1) .AND. LAIM(2).EQ.IAIM(2)) GO TO 490
      DO 485 I=1,4
         AIM(I)=IAIM(I)
         LAIM(I)=IAIM(I)
485   CONTINUE
      GO TO 430
490   IF(PC4.GT.PCINT) THEN
         PCINT=PC4
         NALINT=NUMAIM
         DO 492 I=1,4
            NBMINT(I)=NUMBOM(I)
            AIMINT(I)=AIM(I)
492   CONTINUE
      END IF
493   CONTINUE
      IF(PCINT.GT.PCBEST(NUMB)) THEN
```

```
            PCBEST(NUMB)=PCINT
            NALBST(NUMB)=NALINT
            DO 495 I=1,4
               NBMBST(NUMB,I)=NBMINT(I)
               AIMBST(NUMB,I)=AIMINT(I)
495         CONTINUE
         ELSE
            GO TO 800
         END IF
         IF(NUMB.EQ.4) GO TO 800
C
500      NUMAIM=5
         PC5=0.0
         PCINT=0.0
         LBOMB=(NUMB-3)/2
         DO 593 NBM=1,LBOMB
         SPACE=RUNWID/(NUMAIM+1)
         NUMBOM(1)=NBM
         NUMBOM(5)=NBM
         LBM2=(NUMB-2*NBM)/2
         DO 593 NBM2=1,LBM2
            NUMBOM(2)=NBM2
            NUMBOM(3)=NUMB-2*NBM-2*NBM2
            NUMBOM(4)=NBM2
         DO 510 I=1,NUMAIM
            AIM(I)=I*SPACE
            LAIM(I)=AIM(I)
510      CONTINUE
         IF (SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIM,PC5)
         ELSE
            CALL UNION(NUMAIM,NUMBOM,AIM,PC5)
         END IF
530      STEP1=5.
         AIMI(1)=AIM(1)
         AIMI(3)=CNTR
         AIMI(5)=AIM(5)
         AIMO(1)=AIM(1)
         AIMO(3)=CNTR
         AIMO(5)=AIM(5)
C
C MOVE INNER PAIR OF AIM POINTS IN OR OUT.
C
540      AIMI(2)=AIM(2)+STEP1
         AIMI(4)=AIM(4)-STEP1
         IF (SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIMI,PCI)
         ELSE
            CALL UNION(NUMAIM,NUMBOM,AIMI,PCI)
         END IF
         AIMO(2)=AIM(2)-STEP1
         AIMO(4)=AIM(4)+STEP1
         IF (SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIMO,PCO)
```

```
          ELSE
             CALL UNION(NUMAIM,NUMBOM,AIMO,PCO)
          END IF
          DIF1=PCO-PC5
          DIF2=PC5-PCI
          DIF3=PCI-PC5
          DIF4=PC5-PCO
          IF(DIF1.GT.TOL .AND. DIF2.GT.TOL) THEN
             PC5=PCO
             AIM(2)=AIMO(2)
             AIM(4)=AIMO(4)
          ELSE IF(DIF3.GT.TOL .AND. DIF4.GT.TOL) THEN
             PC5=PCI
             AIM(2)=AIMI(2)
             AIM(4)=AIMI(4)
          ELSE IF(DIF4.GT.TOL .AND. DIF2.GT.TOL) THEN
             IF(STEP1.LE.0.5) GO TO 550
             STEP1=STEP1/2.
             GO TO 540
          END IF
C
C MOVE OUTER PAIR OF AIM POINTS IN OR OUT.
C
550       AIMI(2)=AIM(2)
          AIMI(4)=AIM(4)
          AIMO(2)=AIM(2)
          AIMO(4)=AIM(4)
          STEP2=5.
560       AIMI(1)=AIM(1)+STEP2
          AIMI(5)=AIM(5)-STEP2
          IF (SIMCAL) THEN
             CALL SIM(NUMAIM,NUMBOM,AIMI,PCI)
          ELSE
             CALL UNION(NUMAIM,NUMBOM,AIMI,PCI)
          END IF
          AIMO(1)=AIM(1)-STEP2
          AIMO(5)=AIM(5)+STEP2
          IF (SIMCAL) THEN
             CALL SIM(NUMAIM,NUMBOM,AIMO,PCO)
          ELSE
             CALL UNION(NUMAIM,NUMBOM,AIMO,PCO)
          END IF
          DIF1=PCO-PC5
          DIF2=PC5-PCI
          DIF3=PCI-PC5
          DIF4=PC5-PCO
          IF(DIF1.GT.TOL .AND. DIF2.GT.TOL) THEN
             PC5=PCO
             AIM(1)=AIMO(1)
             AIM(5)=AIMO(5)
          ELSE IF(DIF3.GT.TOL .AND. DIF4.GT.TOL) THEN
             PC5=PCI
             AIM(1)=AIMI(1)
             AIM(5)=AIMI(5)
```

```
      ELSE IF(DIF4.GT.TOL .AND. DIF2.GT.TOL) THEN
         IF(STEP2.LE.0.5) GO TO 570
         STEP2=STEP2/2.
         GO TO 560
      END IF
C
C ROUND OFF AIM POINT LOCATIONS TO NEAREST FOOT.
C
570   DO 580 I=1,5
      IAIM(I)=AIM(I)+0.5
580   CONTINUE
C
C CHECK MOVEMENT FROM LAST ADJUSTMENT.
C
      IF(LAIM(1).EQ.IAIM(1) .AND. LAIM(2).EQ.IAIM(2)) GO TO 590
      DO 585 I=1,5
         AIM(I)=IAIM(I)
         LAIM(I)=IAIM(I)
585   CONTINUE
      GO TO 530
590   IF(PC5.GT.PCINT) THEN
         PCINT=PC5
         NALINT=NUMAIM
         DO 592 I=1,5
            NBMINT(I)=NUMBOM(I)
            AIMINT(I)=AIM(I)
592      CONTINUE
      END IF
593   CONTINUE
      IF(PCINT.GT.PCBEST(NUMB)) THEN
         PCBEST(NUMB)=PCINT
         NALBST(NUMB)=NALINT
         DO 595 I=1,5
            NBMBST(NUMB,I)=NBMINT(I)
            AIMBST(NUMB,I)=AIMINT(I)
595      CONTINUE
      END IF
C
C
800   DIF=PCBEST(NUMB)-PCSTAR
      IF(DIF.GE.TOL) GO TO 1000
C
900   CONTINUE
C
C FOUND THE BEST STRATEGY TO CLOSE THE RUNWAY.
C
1000  PC=PCBEST(NUMB)
      RETURN
      END
      SUBROUTINE RESLTS(NPERCT,NUMBER,PCLEST)
      DIMENSION NPERCT(20)
      COMMON /INPUT/ RUNLEN,RUNWID,TOLEN,TOWID,W,CEP
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
      COMMON /BEST/ PCBEST(20),NALBST(20),NBMBST(20,5),AIMBST(20,5)
C
      PRINT 10,RUNLEN,RUNWID
      PRINT 20,TOLEN,TOWID
      PRINT 30
      PRINT 40,W
      PRINT 50,CEP
      PRINT 60,PCLEST
      PRINT 70,NUMBER
      PRINT 80
      PRINT 90
      DO 200 I=1,NCUTS
      NUMB=NPERCT(I)
      DO 200 J=1,NALBST(NPERCT(I))
         PRINT 100,CUTLOC(I),AIMBST(NUMB,J),NBMBST(NUMB,J)
200   CONTINUE
10    FORMAT('1RUNWAY',13X,F8.0,' BY ',F8.0,' FEET')
20    FORMAT('0MIN LAUNCH WINDOW  ',F8.0,' BY ',F8.0,' FEET')
30    FORMAT('0WEAPON CHARACTERISTICS')
40    FORMAT('0',5X,'YIELD  ',F7.2,' POUNDS')
50    FORMAT('0',5X,'CEP  ',3X,F4.0,'   FEET')
60    FORMAT('0PROBABILITY OF CLOSURE  ',F4.2)
70    FORMAT('0TOTAL NUMBER OF WEAPONS  ',I3)
80    FORMAT('0AIM POINTS',7X,'NUMBER OF')
90    FORMAT('0(LENGTH, WIDTH)',2X,'WEAPONS',/)
100   FORMAT(' ',F7.2,F8.2,4X,I2)
      RETURN
      END
      SUBROUTINE NUMSET(NPERCT,NUMBER,PCLEST)
      DIMENSION NPERCT(20)
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
      COMMON /BEST/ PCBEST(20),NALBST(20),NBMBST(20,5),AIMBST(20,5)
C
      PCSTAR=1.0
      NREM=MOD(NUMBER,NCUTS)
      NUMB=NUMBER/NCUTS
      NUMB1=NUMB+1
      DO 100 I=1,NCUTS
         NPERCT(I)=NUMB
100   CONTINUE
      CALL SEARCH(NUMB,NUMB1,NUMB1,PC)
      DO 200 I=1,NREM
         NPERCT(I)=NPERCT(I)+1
200   CONTINUE
      PCLEST=1.0
      DO 300 I=1,NCUTS
         PCLEST=PCLEST*PCBEST(NPERCT(I))
300   CONTINUE
      RETURN
      END
      SUBROUTINE INDEP
      DIMENSION NUMBOM(20),AIMLOC(20),CUTLCI(20)
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
      COMMON SIMCAL
      LOGICAL SIMCAL
C
      PCIND=1.0
      NUMB=0
      PRINT 100
      PRINT 200
      PRINT 300
      PRINT 400
      READ*,ICUTS
      IF(ICUTS.LT.NCUTS)THEN
         PRINT 800,NCUTS
         PRINT 810,ICUTS
         RETURN
      ELSE IF(ICUTS.GT.NCUTS)THEN
         PRINT 800,NCUTS
         PRINT 820,ICUTS
         RETURN
      END IF
      DO 20 I=1,ICUTS
         READ*,CUTLCI(I),NUMAIM
         PRINT 500,CUTLCI(I)
         IF (I.EQ.1) THEN
            DIST=CUTLCI(I)
         ELSE
            DIST=CUTLCI(I)-CUTLCI(I-1)
         END IF
         IF(DIST.GT.SRL) THEN
            PRINT 900,SRL
            RETURN
         END IF
         DO 10 J=1,NUMAIM
            READ*,AIMLOC(J),NUMBOM(J)
            PRINT 600,AIMLOC(J),NUMBOM(J)
            NUMB=NUMB+NUMBOM(J)
10       CONTINUE
         IF(NUMB.EQ.1)THEN
            CALL PRONE(NUMAIM,NUMBOM,AIMLOC,PC)
         ELSE IF (SIMCAL) THEN
            CALL SIM(NUMAIM,NUMBOM,AIMLOC,PC)
         ELSE IF (.NOT. SIMCAL) THEN
            CALL UNION(NUMAIM,NUMBOM,AIMLOC,PC)
         END IF
         PCIND=PCIND*PC
20    CONTINUE
      PRINT 700,PCIND
100   FORMAT('1INDEPENDENT ANALYSIS')
200   FORMAT('0INDEPENDENT STRATEGY:')
300   FORMAT('0CUT',T12,'AIM POINT',T24,'NUMBER PER')
400   FORMAT(' LOCATION',T12,'LOCATION',T24,'AIM POINT')
500   FORMAT('0',F8.0)
600   FORMAT(' ',T14,F8.2,T27,I3)
700   FORMAT('0PROBABILITY OF RUNWAY CLOSURE IS ',F4.2)
800   FORMAT('0 *** ERROR NUMBER OF CUTS MUST EQUAL ',I4)
```

```
810   FORMAT(' ',I4,' CUTS ARE TOO FEW TO GUARENTEE RUNWAY CLOSURE')
820   FORMAT(' ',I4,' CUTS ARE MORE THAN THE REQUIRED MINIMUM NUMBER OF
     1CUTS')
900   FORMAT('0 *** ERROR DISTANCE BETWEEN CUT LOCATIONS IS TOO LARGE TO
     1 CLOSE RUNWAY.',/,' THE MAXIMUM DISTANCE IS',F10.2,' FEET ALONG TH
     1E RUNWAY')
      RETURN
      END
      SUBROUTINE EVAL(NPERCT,NUMBER,PCLEST)
      DIMENSION NPERCT(20)
      COMMON /BEST/ PCBEST(20),NALBST(20),NBMBST(20,5),AIMBST(20,5)
      COMMON /RWAY/ SRL,PCSTAR,NCUTS,CUTLOC(20)
C
      NUMBER=0
      PCLEST=1.0
      DO 100 I=1,NCUTS
         NUMBER=NUMBER+NPERCT(I)
         PCLEST=PCLEST*PCBEST(NPERCT(I))
100   CONTINUE

      RETURN
      END
```

Vita

Howard Mitsugi Hachida was born on 15 September, 1953 in Honolulu, Hawaii to Mr. and Mrs. Stanley T. Hachida. After graduating from Kaimuki High School in 1971, Howard went on to earn his Bachelor of Arts degree in Mathematics from the University of Hawaii in June 1977. He then entered the USAF Officers Training School, San Antonio, TX where he was commissioned into the United States Air Force as a Second Lieutenant on 9 November 1977. Immediately following his commission into the Air Force, Howard was assigned to the Tactical Fighter Weapons Center at Nellis AFB, NV, where he worked as an Operations Analyst for Operation Red Flag. He also worked as an analyst for the many operational tests that were conducted through the Tactical Fighter Weapons Center. In June 1981 he entered the School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH.

Permanent address: 2445 Holomua Place
Honolulu, Hawaii 96816

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
AFIT/GOR/OS/82D-6

2. GOVT ACCESSION NO.
AD-A124 665

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
A COMPUTER MODEL TO AID THE PLANNING OF RUNWAY ATTACKS

5. TYPE OF REPORT & PERIOD COVERED
MS Thesis

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Howard M. Hachida
Capt USAF

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Air Force Institute of Technology(AFIT/EN)
Wright-Patterson AFB OH 45433

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

12. REPORT DATE
December 1982

13. NUMBER OF PAGES
105

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release: IAW AFR 190-17.
[illegible]
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

19 JAN 1983

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Attacking Strategy
Probabilitv of Runway Closure
Minimum Launch Window

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer program to aid the planning of runway attacks is developed. Conventional, individually targeted weapons are used against non-reinforced concrete runways. The program has two main sections. The first section evaluates any attack strategy, based on independent cuts along the runway, with each cut specified in terms of number of aim points, number of weapons per aim point, and aim point locations. The second section

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20:

searches for the "best" strategy which uses the least number of weapons to achieve an overall probability of runway closure equal to or greater than a user specified level.

The program operates in three modes. The mode 1 program returns the fewest number of weapons and the "best" strategy in order to meet or exceed a user defined level of runway closure. Mode 2 allows the user to specify a fixed number of weapons instead of a level of runway closure, and the program returns the highest probability of runway closure and the "best" strategy to use with the fixed number of weapons. Finally, mode 3 allows the user to completely specify a strategy in terms of number of cuts, cut locations, number of aim points per cut, number of weapons per aim point and locations.

A